University of Alberta

Advances on Design and Analysis of Mesh-Restorable Networks

by

John Ernest Doucette

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the

requirements for the degree of *Doctor of Philosophy*

Department of *Electrical and Computer Engineering*

Edmonton, Alberta
Spring 2005

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

0-494-08227-5

Your file *Votre référence*
ISBN:
Our file *Notre reference*
ISBN:

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

"Mr. Watson, come here, I want you!"

– Alexander Graham Bell (1847 – 1922)

For my mother, Jeannette Warner-Doucette.

All that is good and right in my life is a direct result of your love and support.

ABSTRACT

With the tremendous amount of data transmitted over the world's telecommunication systems and our growing reliance on their continued successful operation, network survivability has never been more important. Yet virtually every day, subscribers somewhere still endure service interruptions that affect their businesses, financial systems, phone systems, or any of a myriad other possible disruptions, up to and including their own personal health and safety, due to unforeseen failures of our networks. The work discussed in this thesis presents new techniques for the optimal design and analysis of network architectures that embody survivability mechanisms to ensure that any communications affected by such failures are restored quickly, efficiently, and inexpensively.

The main outcomes of the research presented are fourfold. First we provide a thorough discussion and analysis of the common mesh network survivability mechanisms; 1+1 automatic protection switching, span restoration, *p*-cycle restoration, shared backup path protection, and path restoration. We next introduce the meta-mesh concept, which is a form of span restoration suited to sparse networks that contain chains of degree-2 nodes. By targeting the loop-back spare capacity required within these chains by working traffic that flows entirely through them, meta-mesh restoration is able to provide substantial savings over conventional span restoration. We then address the problem of jointly optimizing a network's topology as well as the working and restoration routing within it. We show that the complete problem is quite onerous to solve for anything but a very small network, and so we also develop a three-step heuristic that in most cases is actually able to outperform the complete problem in terms of both runtime and solution quality. Finally, we introduce node-inclusive span restoration, which is a completely new form of network restoration based on span restoration, but fundamentally part of the way towards path restoration. We show that node-inclusive span restoration is also capable of node-failure restoration, and in general, most network demands are fully restorable in the event of any node failure with little or no extra spare capacity. At the same time, spare capacity requirements of node-inclusive span restoration is shown to approach path restoration, particularly for highly connected networks.

The insights gained by the work of this thesis add to the growing understanding of the various issues related to network survivability, and has the potential to lower the costs for network operators, while simultaneously providing new options in the planning and development of their future networks. Implementation of design principles from this work will also lead to more reliable communication systems that are less vulnerable to equipment failures or attacks, and might eventually help to eliminate service interruptions that we all tolerate as a necessary aspect of modern telecommunication systems.

ACKNOWLEDGEMENTS

While mine is the name on the first page of this thesis, the work herein would not have been possible if it weren't for the guidance, collaboration, encouragement, support, and friendship provided to me by a great many remarkable people.

First, I thank my supervisor and mentor, Dr. Wayne D. Grover. It is due to his excellent ideas, guidance, and direction that I have been able to grow as an engineer and produce the thesis you now hold. Thanks Wayne, for showing me the value of always demanding the best in my work. I fear I owe you more than I'll ever be able to repay.

I am also grateful to TRLabs for providing me with generous financial support and a professional research environment, without either of which I would not have been able to complete this thesis. The contributions of TRLabs' staff and students are also greatly appreciated. I especially thank Dr. Roger Pederson for his leadership at TRLabs' helm and Linda Richens for all of her administrative support and for her always friendly and helpful demeanour. The discussions and close collaboration with other Network Systems students were particularly valuable, and among them I especially thank Anthony Sack, Dr. Matthieu Clouqueur, Adil Kodian, Aden Grue, Dion Leung, and Gangxiang Shen. In addition, I can single out several other colleagues, past and present, who have also been largely responsible for creating such a friendly office environment: Dr. Chris Haugen, Trevor Allen, David Clegg, Hue Nguyen, Tom Clement, Dr. Vinod Ratti, Robert Elliott, Chris Fields, Demetrios Stamatelakis, Dr. Dave Morley, Travis Robinson, and Ernest Siu. I sincerely hope that any future workplace I find myself in will have people like you there.

I am also grateful to my Ph.D. examining committee members, Dr. Mike MacGregor, Dr. Bruce Cockburn, Dr. Witold Krzymień, and Dr. Lorne Mason, whose helpful comments have greatly improved this thesis. I am particularly grateful to Dr. Mason, who took two days out of his busy schedule to travel from McGill University in Montreal to attend my final oral examination in person.

Finally, for helping to keep me grounded, and being a constant source of encouragement and support, I am forever indebted to the best friends and family a guy could ever have. I especially thank Brian and Sonya, Paul and Sarda, Katja, Darcy, and Tara, as well as my brothers, Jim and Scott, and my father, Arthur. More than anyone, however, it is my mother, Jeannette, to whom I owe the greatest debt and thanks. Your constant and unconditional love and support, and your ability to handle both good fortune and adversity with the same happy outlook are an inspiration. It is you who taught me that if I always make the most of what I have and always do my best, then I will always succeed. In tribute to you, I follow that principle as the one key driving force in everything I do; this thesis is a direct result of that. Thank you.

TABLE OF CONTENTS

CHAPTER 1 Introduction 1
- 1.1 Motivation and Objectives 3
- 1.2 Thesis Outline 3

CHAPTER 2 Mathematical Definitions and Notation 7
- 2.1 Set Theory 7
  - 2.1.1 Sets and Elements 7
  - 2.1.2 Set Operations 8
- 2.2 Graph Theory 9
- 2.3 Operations Research Terminology and Notation 15
  - 2.3.1 Solving Linear Programming Problems 17
  - 2.3.2 Branch and Bound Heuristic for Solving ILPs 17

CHAPTER 3 Optical Transport Networks 23
- 3.1 Introduction to Transport Networks 23
  - 3.1.1 Transport Network Partitioning 24
  - 3.1.2 OSI Transport Layer 25
  - 3.1.3 Transport Networking Technologies 26
- 3.2 SONET 28
- 3.3 Wavelength Division Multiplexing 31
  - 3.3.1 Lightpath Routing 31
- 3.4 Network Demands 32

CHAPTER 4 Network Survivability 33
- 4.1 Basic Concepts 33
  - 4.1.1 Defining Failure 34
  - 4.1.2 Post-Failure 35
- 4.2 Automatic Protection Switching 35
- 4.3 Survivable Ring Networks 37
  - 4.3.1 Uni-directional Path-Switched Rings 37
  - 4.3.2 Bi-directional Line-Switched Rings 38
- 4.4 Mesh Network Survivability 40
  - 4.4.1 Span Restoration 41
  - 4.4.2 Shared Backup Path Protection 43
  - 4.4.3 Path Restoration 44
  - 4.4.4 *p*-Cycles 46
- 4.5 Protected Working Capacity Envelope 48
- 4.6 Restoration versus Protection 50
- 4.7 Redundancy as a Measure of Network Efficiency 51
- 4.8 Other Issues and Challenges in Network Survivability 54

CHAPTER 5 Mesh-Restorable Transport Network Design and Optimization 57
- 5.1 Arc-Path ILP Formulations 59
  - 5.1.1 Span Restoration 60
  - 5.1.2 Shared Backup Path Protection 65
  - 5.1.3 Path Restoration 70
  - 5.1.4 *p*-Cycles 73
- 5.2 Pre-Processing for Route Enumeration 75

5.2.1 Eligible Route Hop or Distance Limits 76
5.3 Heuristic and Algorithmic Approaches 77
5.3.1 ILP-Based Heuristics 80
CHAPTER 6 Experimental Set-up and Benchmarking 83
6.1 Network Topology Models 83
6.2 Demand Models 85
6.3 Network Design and Solution Methods 87
6.3.1 Computational Aspects 87
6.3.2 Validation Considerations 90
6.4 Comparative Aspects of Basic Model Benchmarks 91
6.4.1 Total Capacity Costs 91
6.4.2 Spare Capacity Costs 99
6.4.3 Working Capacity Costs 104
6.4.4 Capacity Cost Redundancies 109
CHAPTER 7 Meta-Mesh Network Design 115
7.1 Chains and Loop-Back 117
7.2 Breaking Down Working Capacity Into Local and Express Components 119
7.3 The Meta-Mesh 121
7.3.1 Augmenting the Topology With Logical Chain Bypasses 123
7.4 Meta-Mesh Design Model 124
7.5 Experimental Study Method 127
7.6 Results and Discussion 128
7.6.1 Meta-Mesh Capacity Costs 128
7.6.2 Meta-Mesh versus Span-Restorable JCA 142
7.7 Closing Remarks 160
7.7.1 Future Directions 161
CHAPTER 8 Topological Transport Network Design 163
8.1 Prior Work 167
8.1.1 Fixed Charge Plus Routing 170
8.2 Mesh Topology, Routing, and Sparing 177
8.2.1 MTRS ILP Formulation 178
8.2.2 Initial MTRS Testing 182
8.3 Three-Stage Heuristic Solution Method 199
8.3.1 Step W1 – Working-Only Fixed Charge plus Routing 202
8.3.2 Step S2 – Reserve Network Fixed Charge and Sparing 203
8.3.3 Step J3 – Restricted MTRS for Final Topology and Capacity 206
8.3.4 Additional Considerations and Optional Bounds 207
8.4 Experimental Study Method 208
8.5 Results and Discussion 208
8.5.1 Relaxations for Lower Bounds on Optimal Reference Solutions 219
8.6 Closing Remarks 220
8.6.1 Future Directions 222
CHAPTER 9 Node-Inclusive Span Restoration 225
9.1 Basic Concepts 226
9.1.1 Operational Considerations and Other Details 230
9.2 NISR Design Models 234
9.2.1 Span-Failure Restoration 234
9.2.2 Node-Failure Restoration 237

9.2.3 Combined Span-Failure and Node-Failure Restoration 241
9.3 Experimental Study Method 243
9.4 Results and Discussion 244
9.4.1 NISR-S Capacity Costs 244
9.4.2 NISR-S Redundancy 262
9.4.3 Using NISR for Node-Failure Restoration 265
9.5 Closing Remarks 272
9.5.1 Considerations for Dynamic Service Provisioning 274
9.5.2 Future Directions 275
CHAPTER 10 Closing Discussion 277
10.1 Summary of Thesis 277
10.1.1 Main Contributions 278
10.2 Other Contributions of Ph.D. Work 279
10.2.1 Journal Papers 280
10.2.2 Peer-Reviewed Conference Papers 280
10.2.3 Book Chapter 282
10.2.4 Patents Pending 282
10.2.5 Technical Reports and Presentations 282
References 285
APPENDIX A Network Topology Files 295
A.1 15n30s1 Master Network 295
A.1.1 15n30s1 Network Family Members' Span Listings 296
A.2 20n40s1 Network Family 297
A.2.1 20n40s1 Network Family Members' Span Listings 298
A.3 25n50s1 Network Family 300
A.3.1 25n50s1 Network Family Members' Span Listings 301
A.4 30n60s1 Network Family 304
A.4.1 30n60s1 Network Family Members' Span Listings 305
A.5 35n70s1 Network Family 309
A.5.1 35n70s1 Network Family Members' Span Listings 311
A.6 40n80s1 Network Family 315
A.6.1 40n80s1 Network Family Members' Span Listings 317
APPENDIX B Network Demand Files 323
B.1 15n30s1 Network Family 323
B.2 20n40s1 Network Family 324
B.3 25n50s1 Network Family 326
B.4 30n60s1 Network Family 328
B.5 35n70s1 Network Family 331
B.6 40n80s1 Network Family 335

LIST OF TABLES

Table 3.1 – Typical SONET data rates. 31
Table 8.1 – Edge-to-unit-capacity cost ratios for test case network families...........184
Table 8.2 – MTRS solutions of all six master networks, each with three different edge-to-unit-capacity cost ratios. 189
Table 8.3 – MTRS solutions of 15n30s1-Full and 20n40s1-Full full-mesh networks, each with three different edge-to-unit-capacity cost ratios. 191
Table 8.4 – Heuristic solutions of all six master networks, each with three different edge-to-unit-capacity cost ratios. 210
Table 8.5 – Heuristic solutions of the full-mesh versions of the 15-node, 20-node, 25-node, and 30-node master networks, each with three different edge-to-unit-capacity cost ratios. 217

LIST OF FIGURES

Figure 2.1 – Examples of graphs. 9
Figure 2.2 – Examples of (a) isomorphic graphs and (b) homeomorphic graphs......11
Figure 2.3 – Examples of (a) connected, (b) two-connected, and (c) bi-connected graphs. 14
Figure 2.4 – Examples of (a) a tree, (b) a forest, and (c) a spanning tree (in bold)...15
Figure 2.5 – Illustrating the first steps of the branch-and-bound heuristic..................18
Figure 2.6 – Illustration of branch-and-bound heuristic with node 2 solved. 19
Figure 2.7 – Illustration of branch-and-bound heuristic with 5 nodes solved. 20
Figure 2.8 – Illustration of branch-and-bound heuristic with 7 nodes solved. 21
Figure 2.9 – Illustration of branch-and-bound heuristic with all nodes solved...........22
Figure 3.1 – Examples of service layer stacking arrangements in modern transport networks. 24
Figure 3.2 – Partitioned view of a modern transport network. 25
Figure 3.3 – The OSI model and how it relates to the transport network...................26
Figure 3.4 – A simple optical cross-connect (OXC)..27
Figure 3.5 – WaveStar™ LambdaRouter Optical Cross-Connect Switch: with a standard sewing needle to show scale (left) and a close-up of one mirror partially tilted (right), copyright 2004, Lucent Technologies...............................27
Figure 3.6 – A simple optical add/drop multiplexer (OADM).......................................28
Figure 3.7 – The structure of the SONET STS-1 frame...30
Figure 4.1 – Illustrations of 1:N automatic protection switching.................................36
Figure 4.2 – Basic operation of a UPSR (a) before failure, and (b) after failure.........38
Figure 4.3 – Basic operation of a BLSR (a) before failure, and (b) after failure. 39
Figure 4.4 – An example of span restoration. 41
Figure 4.5 – An illustration of shared backup path protection....................................43
Figure 4.6 – An illustration of path restoration with stub-release. 45
Figure 4.7 – An illustration of *p*-cycle restoration. 47
Figure 4.8 – Spans protected by (a) an eight-span ring and (b) a comparable eight-span *p*-cycle. 48
Figure 4.9 – The basis for a derivation of a topological lower bound on redundancy in a span-restorable network. 53
Figure 6.1 – Network topologies of the (a) 15n30s1, (b) 20n40s1, (c) 25n50s1, (d) 30n60s1, (e) 35n70s1, and (f) 40n80s1 master networks. 84
Figure 6.2 – Demonstrating the routing infeasibility problem in SBPP networks. 89
Figure 6.3 – Normalized total capacity costs for the 15n30s1 network family...........92
Figure 6.4 – Normalized total capacity costs for the 20n40s1 network family...........92
Figure 6.5 – Normalized total capacity costs for the 25n50s1 network family...........93
Figure 6.6 – Normalized total capacity costs for the 30n60s1 network family...........93
Figure 6.7 – Normalized total capacity costs for the 35n70s1 network family...........94
Figure 6.8 – Normalized total capacity costs for the 40n80s1 network family...........94
Figure 6.9 – Normalized spare capacity costs for the 15n30s1 network family. 99
Figure 6.10 – Normalized spare capacity costs for the 20n40s1 network family.....100
Figure 6.11 – Normalized spare capacity costs for the 25n50s1 network family.....100
Figure 6.12 – Normalized spare capacity costs for the 30n60s1 network family.....101

Figure 6.13 – Normalized spare capacity costs for the 35n70s1 network family.....101
Figure 6.14 – Normalized spare capacity costs for the 40n80s1 network family.....102
Figure 6.15 – Normalized working capacity costs for the 15n30s1 network family. 106
Figure 6.16 – Normalized working capacity costs for the 20n40s1 network family. 106
Figure 6.17 – Normalized working capacity costs for the 25n50s1 network family. 107
Figure 6.18 – Normalized working capacity costs for the 30n60s1 network family. 107
Figure 6.19 – Normalized working capacity costs for the 35n70s1 network family. 108
Figure 6.20 – Normalized working capacity costs for the 40n80s1 network family. 108
Figure 6.21 – Capacity cost redundancy for the 15n30s1 network family...............110
Figure 6.22 – Capacity cost redundancy for the 20n40s1 network family...............111
Figure 6.23 – Capacity cost redundancy for the 25n50s1 network family...............111
Figure 6.24 – Capacity cost redundancy for the 30n60s1 network family...............112
Figure 6.25 – Capacity cost redundancy for the 35n70s1 network family...............112
Figure 6.26 – Capacity cost redundancy for the 40n80s1 network family...............113
Figure 7.1 – The Level 3 Communications, Inc. USA backbone network [75]. 116
Figure 7.2 – The amount of spare capacity on a chain is determined by the size of the largest working capacity total on any span within the chain. 119
Figure 7.3 – Working capacity in a chain supports local and express flow of working routing. 120
Figure 7.4 – Allowing express flows to fail all the way back to the anchor nodes reduces loop-back spare capacity required within the chain. 121
Figure 7.5 – The meta-mesh of the Level 3 Communications, Inc. USA backbone network...122
Figure 7.6 – Test case 25n50s1-26s network topology. 130
Figure 7.7 – Meta-Mesh normalized total capacity costs for the 15n30s1 network family...131
Figure 7.8 – Meta-Mesh normalized total capacity costs for the 20n40s1 network family...131
Figure 7.9 – Meta-Mesh normalized total capacity costs for the 25n50s1 network family...132
Figure 7.10 – Meta-Mesh normalized total capacity costs for the 30n60s1 network family...132
Figure 7.11 – Meta-Mesh normalized total capacity costs for the 35n70s1 network family...133
Figure 7.12 – Meta-Mesh normalized total capacity costs for the 40n80s1 network family...133
Figure 7.13 – Meta-Mesh normalized spare capacity costs for the 15n30s1 network family...135
Figure 7.14 – Meta-Mesh normalized spare capacity costs for the 20n40s1 network family...135
Figure 7.15 – Meta-Mesh normalized spare capacity costs for the 25n50s1 network family...136
Figure 7.16 – Meta-Mesh normalized spare capacity costs for the 30n60s1 network family...136
Figure 7.17 – Meta-Mesh normalized spare capacity costs for the 35n70s1 network family...137
Figure 7.18 – Meta-Mesh normalized spare capacity costs for the 40n80s1 network family...137

Figure 7.19 – Meta-mesh capacity cost redundancy for the 15n30s1 network family. 139
Figure 7.20 – Meta-mesh capacity cost redundancy for the 20n40s1 network family. 140
Figure 7.21 – Meta-mesh capacity cost redundancy for the 25n50s1 network family. 140
Figure 7.22 – Meta-mesh capacity cost redundancy for the 30n60s1 network family. 141
Figure 7.23 – Meta-mesh capacity cost redundancy for the 35n70s1 network family. 141
Figure 7.24 – Meta-mesh capacity cost redundancy for the 40n80s1 network family. 142
Figure 7.25 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 15n30s1 network family. 145
Figure 7.26 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 20n40s1 network family. 145
Figure 7.27 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 25n50s1 network family. 146
Figure 7.28 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 30n60s1 network family. 146
Figure 7.29 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 35n70s1 network family. 147
Figure 7.30 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 40n80s1 network family. 147
Figure 7.31 – Meta-mesh total capacity cost reduction relative to span restoration JCA versus average nodal degree for all 163 test case networks. 150
Figure 7.32 – Meta-mesh total capacity cost reduction relative to span restoration JCA versus express flow capacity for all 163 test case networks. 150
Figure 7.33 – Meta-mesh total capacity cost reduction relative to span restoration JCA versus the number of chain spans for all 163 test case networks. 153
Figure 7.34 – Meta-mesh total capacity cost reduction relative to span restoration JCA versus the percentage of chain spans for all 163 test case networks. 154
Figure 7.35 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 15n30s1 network family. 157
Figure 7.36 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 20n40s1 network family. 157
Figure 7.37 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 25n50s1 network family. 158
Figure 7.38 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 30n60s1 network family. 158
Figure 7.39 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 35n70s1 network family. 159
Figure 7.40 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 40n80s1 network family. 159
Figure 8.1 – Breakdown of sample complete network design costs for the 40n80s1 network family. 166
Figure 8.2 – Complete network design costs for the 15n30s1 network family with various edge-to-unit-capacity cost ratios. 185

Figure 8.3 – Complete network design costs for the 20n40s1 network family with various edge-to-unit-capacity cost ratios........185
Figure 8.4 – Complete network design costs for the 25n50s1 network family with various edge-to-unit-capacity cost ratios........186
Figure 8.5 – Complete network design costs for the 30n60s1 network family with various edge-to-unit-capacity cost ratios........186
Figure 8.6 – Complete network design costs for the 35n70s1 network family with various edge-to-unit-capacity cost ratios........187
Figure 8.7 – Complete network design costs for the 40n80s1 network family with various edge-to-unit-capacity cost ratios........187
Figure 8.8 – MTRS solutions of the 15n30s1 master network superimposed on the complete network design costs for the entire network family. 192
Figure 8.9 – MTRS solutions of the 20n40s1 master network superimposed on the complete network design costs for the entire network family. 192
Figure 8.10 – MTRS solutions of the 25n50s1 master network superimposed on the complete network design costs for the entire network family. 193
Figure 8.11 – MTRS solutions of the 30n60s1 master network superimposed on the complete network design costs for the entire network family. 193
Figure 8.12 – Optimal MTRS and reference solution network graphs for 15n30s1-based designs with three different edge-to-unit-capacity cost ratios. 196
Figure 8.13 – Optimal MTRS and reference solution network graphs for 20n40s1-based designs with three different edge-to-unit-capacity cost ratios. 196
Figure 8.14 – Optimal MTRS and reference solution network graphs for 25n50s1-based designs with three different edge-to-unit-capacity cost ratios. 197
Figure 8.15 – Optimal MTRS and reference solution network graphs for 30n60s1-based designs with three different edge-to-unit-capacity cost ratios. 197
Figure 9.1 – Custodial nodes in (a) span restoration, and (b) path restoration. 226
Figure 9.2 – Illustrating node-inclusive span restoration for span failures. 228
Figure 9.3 – Illustrating node-inclusive span restoration for node failures. 229
Figure 9.4 – How NISR responds to specific failures. 231
Figure 9.5 – NISR restoring multiple lightpaths simultaneously........233
Figure 9.6 – Special cases of NISR restoration........234
Figure 9.7 – With NISR, 100% node-failure restoration doesn't necessarily imply 100% span-failure restoration. 242
Figure 9.8 – NISR-S SCA normalized spare capacity costs for the 15n30s1 network family. 245
Figure 9.9 – NISR-S SCA normalized spare capacity costs for the 20n40s1 network family. 246
Figure 9.10 – NISR-S SCA normalized spare capacity costs for the 25n50s1 network family. 246
Figure 9.11 – NISR-S SCA normalized spare capacity costs for the 30n60s1 network family. 247
Figure 9.12 – NISR-S SCA normalized spare capacity costs for the 35n70s1 network family. 247
Figure 9.13 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 15n30s1 network family. 250

Figure 9.14 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 20n40s1 network family. 250
Figure 9.15 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 25n50s1 network family. 251
Figure 9.16 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 30n60s1 network family. 251
Figure 9.17 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 35n70s1 network family. 252
Figure 9.18 – Scatter plot of the gap ratios of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for all 124 test case networks. 253
Figure 9.19 – Scatter plot of the average working route length (in hops) of shortest path working routing versus average nodal degree for all 124 test case networks. 255
Figure 9.20 – Scatter plot of the percentage of working routes of three hops or less in shortest path working routing versus average nodal degree for all 124 test case networks. 255
Figure 9.21 – Scatter plot of the percentage of working routes of three hops or less in shortest path working routing versus their average length (in hops) for all 124 test case networks. 256
Figure 9.22 – Scatter plot of the gap ratios of NISR-S SCA capacity costs versus average working route lengths (in hops) for all 124 test case networks. 258
Figure 9.23 – Scatter plot of the gap ratios of NISR-S SCA capacity costs versus the percentage of working routes of three hops or shorter for all 124 test case networks. 258
Figure 9.24 – Normalized SCA spare capacity cost linear regression trend lines plotted against average nodal degree for the 25n50s1 network family. 260
Figure 9.25 – Normalized SCA spare capacity cost linear regression trend lines plotted against average working route length for the 25n50s1 network family. 261
Figure 9.26 – Normalized SCA spare capacity cost linear regression trend lines plotted against the percentage of three-hop working routes for the 25n50s1 network family. 261
Figure 9.27 – NISR-S SCA capacity cost redundancy for the 15n30s1 network family. 263
Figure 9.28 – NISR-S SCA capacity cost redundancy for the 20n40s1 network family. 263
Figure 9.29 – NISR-S SCA capacity cost redundancy for the 25n50s1 network family. 264
Figure 9.30 – NISR-S SCA capacity cost redundancy for the 30n60s1 network family. 264
Figure 9.31 – NISR-S SCA capacity cost redundancy for the 35n70s1 network family. 265
Figure 9.32 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 15n30s1 network family. 266

Figure 9.33 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 20n40s1 network family. 266
Figure 9.34 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 25n50s1 network family. 267
Figure 9.35 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 30n60s1 network family. 267
Figure 9.36 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 35n70s1 network family. 268
Figure 9.37 – Node-failure un-restorability levels (excluding terminating lightpaths) of all 124 test case networks as designed with the NISR-S optimization model. .272

CHAPTER 1

INTRODUCTION

Determining the transmission links, switching, and access equipment required to handle the features and services that millions of telecommunication network users want is an incredibly complex and costly problem. Every year, companies and governments spend billions of dollars on installation and maintenance of their networks, and they expect at least limited assurances on their continued uninterrupted operation. The consequences of failure of just a single cable can be catastrophic and far-reaching, however, affecting business operations, banking, security systems, healthcare, 911 service, air traffic control and more. The advent of high capacity dense wavelength division multiplexing (DWDM) networking makes the proper design of telecommunication networks even more important, and though core backbone transport networks already operate at speeds in the terabit per second range, capacities are expected to continue to double as often as every six months [44] (even in the presence of recent economic challenges faced by the telecommunications industry). The growth of the Internet, the emergence of the so-called "global community," industry and public needs for high reliability, and society's thirst for faster, better, and cheaper communications all place their own demands on existing and future networks.

Despite considerable efforts taken to provide physical protection of cables, FCC statistics show that metro networks annually experience approximately 13 cuts for every 1000 miles of fibre, and long haul networks experience 3 cuts for 1000 miles fibre [115]. Even the lower rate for long haul networks implies a cable cut every four days on average in a typical network with 30,000 route-miles of fibre. In the first four months of 2002 alone, the FCC logged 50 separate network outages throughout the United States [30], with wide-ranging effects and in some cases, very peculiar causes:

- On 13 February 2002 in Yadkinville, NC, town workers severed a Sprint cable while repairing a water line, cutting 52 trunk groups and 13 DS3 links for over 5 hours [31].

- On 19 February 2002, a fire in a Maryland power transformer melted a Verizon fibre cable affecting 5000 customers for over 9 hours [32].
- On 14 March 2002, a contractor accidentally cut functional fibre during removal of retired fibre, cutting 911 service to a part of San Diego for over 4 hours [33].

Canada is not immune to similar network failures. In fact, on the very day this thesis was defended (and purely coincidentally, we can assume), construction workers in Vermilion, AB severed a fibre cable, leaving thousands of residents in Lloydminster, AB without any phone, 911, or cellular service at all for 11 hours [97]. In the last few months alone, CA*net 4 [11] suffered numerous outages:

- On 20 May 2004, a faulty optical amplifier brought down an OC-192 between Vancouver, BC and Victoria, BC for over 10 hours [12].
- On 17 August 2004, a boat anchor cut a CA*net 4 cable off Halifax, NS, and it required 9 days to return an OC-192 between Montreal, PQ and Halifax back to service [12].
- On 15 October 2004, an OC-192 between Winnipeg, MB and Chicago, IL was down for approximately 3 hours due to a "fibre cut" [12].

Clearly, network failures are common, and designing a network that can recover from the failure before an outage occurs has become essential as our networks continue to develop and grow. In recognition of the importance of continued operation of our national telecommunication systems, the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Department of Public Safety and Emergency Preparedness Canada (PSEPC) have even identified the telecommunications system as one of Canada's ten most critical infrastructures. They have also teamed to fund the Joint Infrastructure Interdependencies Research Program (JIIRP), which seeks to "produce new science-based knowledge and practices to better assess, manage, and mitigate risks to Canadians from critical infrastructure," [88].

Perhaps most notably, the Gartner Group predicts, "*Through 2004, large U.S. enterprises will have lost more than $500 million in potential revenue due to network failures that affect critical business functions*," [91]. So the ability of a network to sustain a failure and continue uninterrupted operation is perhaps one of the most critical attributes to consider. Accordingly, network survivability has become an integral part of the design of optical transport networks.

Recent years have seen great progress in survivable networking technologies that enable telecommunication systems to continue operating without disruption, despite the inevitable failure of some network elements. Key network restoration protocols have been studied for at least the last two decades, and every year, the world's telecommunications journals and conference are filled with new research on the topic. The work discussed in this thesis adds to that body of knowledge, and presents new techniques for optimal design and analysis of network architectures that embody such restoration mechanisms.

1.1 MOTIVATION AND OBJECTIVES

With the vast array of choices available to network designers looking to protect their networks against failures, deciding which mechanism to choose and how to implement it can be challenging. The primary contribution of our research is to provide greater understanding and insights into operational and design aspects of mesh-based transport networks, as well as present alternatives to the conventional methods already in use. Our research also aims to reduce the time and effort required to develop complex network designs, and provide an improved comparative appreciation of the various mesh networking options available. The telecommunications industry will benefit from this research by being able to respond more quickly to customer requirements and market fluctuations, and our improvements in design optimality will lead to lower network design costs, passing the advantages on to the public as a whole.

1.2 THESIS OUTLINE

The remainder of this thesis contains a thorough discussion of relevant background material, followed by benchmarking analyses, and finally, comprehensive analyses and discussion of three key advances in the field of survivable networks.

In CHAPTER 2, we introduce set theory, graph theory, and operations research, which will all be used extensively throughout the thesis. CHAPTER 3 gives an overview of optical transport networking and covers the key technologies to which our network design methods are applied. A comprehensive explanation of network survivability methods is given in CHAPTER 4, with an emphasis on mesh networking. CHAPTER 5 develops the mathematical formulations of the network design models corresponding to each of the fundamental survivability mechanisms discussed in CHAPTER 4.

In CHAPTER 6, we present our test networks and demand models used herein, discuss our experimental methods, and provide thorough analyses of all the basic survivability mechanisms, which we thereafter use as comparative benchmarking results for the work of the next three chapters. The analyses carried out in this chapter examines and compares various mesh network restoration and protection mechanisms and design schemes in terms of overall capacity requirements and restoration capacity efficiency. It is based on work in [52] and [54], which are the first such comprehensive studies, and provides useful insights and understanding of how the various mesh schemes behave over varying network connectivity.

In CHAPTER 7, we develop and discuss the concept of meta-mesh networking, which is a new method of designing and implementing span-restoration in sparse (i.e., not richly connected) mesh networks, networks that are often judged to be suitable only for rings. The restoration method developed increases capacity efficiency in sparse networks by targeting chains of degree-2 nodes (that ordinarily require ring-like loop-back and spare capacity allocation), and allowing the restoration mechanism to fail express traffic – traffic that fully transits the chain end-to-end – to the end-nodes of the chain, thereby greatly reducing the loop-back capacity requirements. The result is that capacity efficiency nears that of path restoration (by virtue of restoration routes that extend further back along the working lightpaths), but restoration is by the much simpler span restoration mechanism.

We study topological transport network design methods in CHAPTER 8, where we survey the state of the art of the topic. We then develop an optimal mathematical model and a fast 3-step heuristic approach to solve the full "green-fields" problem where a network is constructed entirely from the ground up. We show that in all but the simplest test cases, the heuristic outperforms the full problem in terms of run-time and is actually able to produce superior network designs when compared to reasonably run-time limited attempts at the full problem.

CHAPTER 9 introduces node-inclusive span restoration (NISR), a new network restoration method that is a modification of the span restoration mechanism such that node-failures are restorable, and is notionally a compromise between span restoration and path restoration. Our analyses show that in highly connected networks, NISR even begins to approximate the capacity efficiency of path restoration. Tests also showed that with NISR, very little additional spare capacity is needed to allow for full node-failure restorability above that needed for span-failure recovery, and that if designed only for the latter, most demands actually experience full node-failure restorability anyway.

We close with a summary discussion in CHAPTER 10, which includes a review of the main contributions of this thesis, as well as other contributions of the Ph.D. work.

CHAPTER 2
MATHEMATICAL DEFINITIONS AND NOTATION

The work presented in this thesis makes use of various concepts of mathematical set theory, graph theory, and operations research. In order to ensure that the reader has a suitable understanding of those areas of study, we provide an overview of the basic concepts, symbols, and notation.

2.1 SET THEORY

2.1.1 SETS AND ELEMENTS

The most fundamental concept in set theory [8] (and the source of its name) is the concept of *sets*, an unordered collection of objects related to each other by their inclusion in the set. The objects sharing membership in a set are called its *elements*, and wholly specify the set. A set's elements can also be considered to enjoy some unique property or group of properties solely as a consequence of belonging to the set. We denote a set S containing elements x, y, and z, as $S=\{x,y,z\}$, and note that since order doesn't matter in sets, this is equivalent to defining the set as $S=\{z,y,x\}$. We can also denote membership of an element x in a set S as $x \in S$. If an element x does not belong to the set S, then $x \notin S$. We can refer to all elements in the set by $\forall x \in S$, where $\forall$ is the *universal quantifier* meaning "for each" (the entire notation can be read as "for each element x having membership in the set S"). The *existential quantifier*, $\exists$, means "there exists a" and | means "such that", so $\forall x \in T | \exists y \in S, x > y$ is read as "for all elements x in the set T such that there exists an element y in the set S where x is greater than y."

When defining a set, we can assume that its elements are drawn from a population of a certain kind of like objects (be they positive integers, spatial coordinates, people, buildings, colours, network nodes, etc.) called *primordial elements*, making up the *base type*, or *primordial set*. An example of a well-known primordial set is the set of all natural numbers, denoted as $\mathbb{N}=\{1,2,3,\ldots,\infty\}$. A primordial element either belongs to a set or it doesn't; there is no concept of duplication of elements in a set.

Two sets S and T are *equal* if they contain exactly the same elements, and are denoted as such by $S = T$. If the two sets do not contain exactly the same elements, then they are not equal, denoted by $S \neq T$. A set T can be defined as a *subset* of set S if it contains only elements that are also in set S. A subset of a set S is often denoted as S' and can be expressed as $S' \subseteq S$, or alternatively, $S \supseteq S'$, where the relation $\subseteq$ is called *inclusion*. If $T \subseteq S$, then S is called a *superset* of T. A subset can be defined as a *proper subset* if $T \subseteq S$ but $T \neq S$, or in other words, T contains only elements of S but not all of them. This is denoted by $T \subset S$ or $S \supset T$, where the relation $\subset$ is called *strict inclusion*.

A *sequence* is similar to a set and uses the same notation, but in sequences, the elements' order matters. For instance, the sequence $S_1 = \{x, y, z\}$ is different from the sequence $S_2 = \{z, y, x\}$. Duplicate elements are allowed in a sequence.

2.1.2 SET OPERATIONS

We define the *union* of two sets $R = \{u, v, w, x\}$ and $S = \{x, y, z\}$ as the set of all elements that appear in either set (or both), denoted as $R \cup S = \{u, v, w, x, y, z\}$. The *intersection* of R and S is the set of all elements that appear in both sets, and is denoted as $R \cap S = \{x\}$. The *difference* of two sets, denoted by $R - S$, is the set of elements in R but not in S, so $R - S = \{u, v, w\}$ and $S - R = \{y, z\}$. The *symmetric difference* of two sets is the set of all elements appearing in either of the sets but not both, and is denoted as $R \Delta S = \{u, v, w, y, z\}$. A set's complement $\bar{S}$ is the set of all primordial elements that are not in the original set S, so if the primordial set is $P = \{s, t, u, v, w, y, z\}$, then $\bar{S} = \{s, t, u, v, w\}$. The *cardinality* of a set is defined as the number of elements in the set and is denoted by $|S|$, so $|R| = 4$ because R has four elements, and $|S| = 3$ because S has three elements. A set Q containing no elements is called an *empty set* and can be denoted by $Q = \varnothing$, $Q = \{\ \}$, or $|Q| = 0$.

2.2 GRAPH THEORY

A graph can best be described as an abstract mathematical entity denoted by $G=(V,E)$, and is composed of two sets, where the set $V=\left\{v_1,v_2,v_3,\ldots,v_{|M|}\right\}$ contains the vertices or points in typically two-dimensional space in the graph, and the set $E=\left\{e_1,e_2,e_3,\ldots,e_{|E|}\right\}$ contains all distinct edges connecting those vertices [9]. An edge e_1 is said to *join* the two vertices v_1 and v_2 it connects, which are called its *end vertices*. An edge is said to be *incident* on its end vertices, and they are said to be incident on it. Two vertices are *adjacent* to each other if they are joined by an edge, as are two edges that share a common end vertex. Graphs can be represented geometrically as well by drawing the vertices as points and edges as lines connecting the vertices as shown in Figure 2.1. In transport networking, the terms vertex and edge are used interchangeably with *node* and *span*, respectively, where nodes contain OXCs, OADMs, IP routers, or other such equipment, and spans are typically lengths of optical fibre cables.

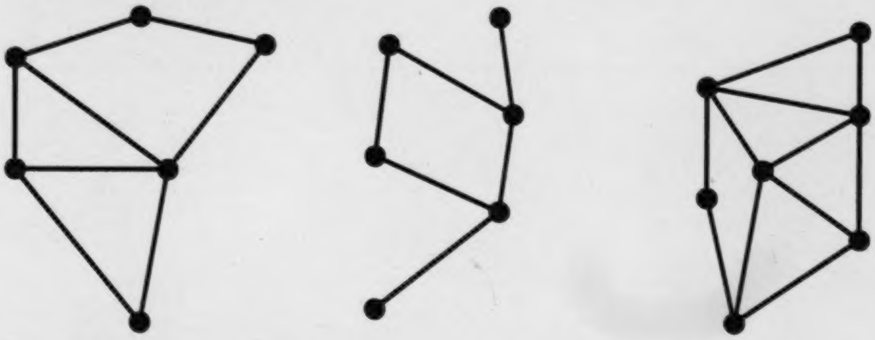


Figure 2.1 – Examples of graphs.

An edge whose two end vertices are actually the same vertex (i.e., the edge starts and ends on a single vertex), is called a *self-loop*. Two edges that share the same pair of end vertices are *parallel*. Any graph that has no parallel edges or self-loops is called a *simple* graph. A graph containing at least one pair of parallel edges (but no self-loops) is called a *multigraph* and the number of parallel edges between a single pair of vertices is that edge's *multiplicity*. A graph that contains at least one self-loop is called a *general* graph.

Normally an edge can be referred to as a set $e_1 = \{v_1, v_2\}$ but it can also be referred to as an *ordered pair* $e_1 = \{v_1, v_2\}$ (same notation) if the v_1 is thought of as the *origin* of the edge and v_2 is the *destination* or *terminus* of the edge. An ordered pair is simply a sequence with two elements. In such a case, the edge is considered to be *directed* and is drawn with an arrowhead pointing to the terminus. A graph with at least one directed edge is a *directed* graph or *digraph*, and a graph with no directed edges is called an *undirected* graph. A vertex that is the origin of all edges incident on it is called a *source* and a vertex that is the terminus of all edges incident on it is called a *sink*. The transport networks modelled in this thesis are represented by undirected graphs since in such networks, transmission capacity on each span is bi-directional (and symmetric).

A graph's *order* is the number of vertices $|V|$ in the graph, and a graph is *complete* if each pair of distinct vertices forms an edge (and by extension, each vertex is adjacent to every other). A complete graph of order n is denoted by K_n and will contain ${}^{n}C_2 = \frac{n(n-1)}{2}$ edges. The number of edges incident on a vertex v is called the vertex's *valence* or *degree* and is denoted by d_v. Vertices with degree equal to 1 are called *stubs*. The average degree of all vertices in a graph can easily be calculated by $\bar{d} = \frac{2 \cdot |E|}{|V|}$, and is called the *network average nodal degree* in transport networking terminology.

A graph $G' = (V', E')$ is called a *subgraph* of graph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$, and we can say $G' \subseteq G$. Two graphs $\tilde{G} = (\tilde{V}, \tilde{E})$ and $G = (V, E)$ are *isomorphic* if a one-to-one correspondence can be made between the vertices in each graph, as well as the edges that connect them. In other words, (a) each vertex in $\tilde{G} = (\tilde{V}, \tilde{E})$ corresponds directly to a specific vertex in $G = (V, E)$ and vice versa, and (b) each edge connecting a pair of vertices in $\tilde{G} = (\tilde{V}, \tilde{E})$ corresponds directly to the specific edge in $G = (V, E)$ connecting the corresponding pair of vertices. Two graphs that are isomorphic must each contain the same number of vertices and edges, but this property alone does not guarantee that two graphs are isomorphic. If a vertex of degree 2 in $G=(V,E)$ is removed and the two edges incident on it are replaced by a single edge whose end vertices are the remaining end vertices of the original pair of edges, the resulting graph $\tilde{G}=(\tilde{V},\tilde{E})$ is a *homeomorphism* of the first graph. Isomorphic and homeomorphic graphs are illustrated in Figure 2.2.

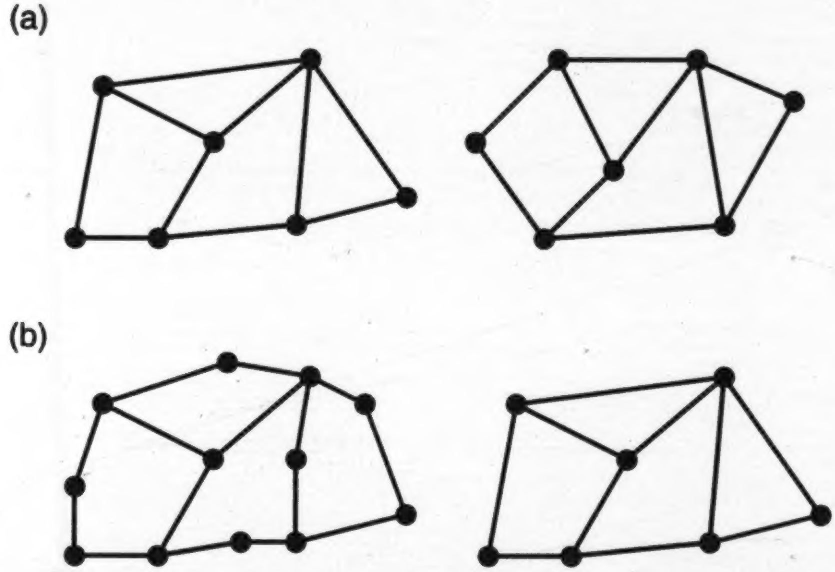


Figure 2.2 – Examples of (a) isomorphic graphs and (b) homeomorphic graphs.

Many graphs can be represented by a drawing on a plane such that no edges will intersect each other except where they meet at end vertices. Such a graph is considered to be *planar*. Any graph that is isomorphic with a planar graph is planar. It turns out that if we represent real transport networks as graphs, most of them are planar. Planarity imparts some interesting properties on a graph. For instance, a planar graph must have at least one vertex whose degree is 5 or less, and it can be divided into $r=|E|-|V|+2$ non-overlapping regions or faces [9]; this can be useful knowledge when designing ring or *p*-cycle networks.

A graph whose edges each have a number w_e associated with it is called a *weighted* graph, and the number w_e is the *weight* of the edge. A weighted graph whose edge weights represent capacities is called a *capacitated* graph. We typically use capacitated graphs to represent transport networks where the edges each have multiple weights associated with them to represent the bandwidths or wavelength capacities, distances, costs, etc. In transport networking, we refer to a single unit of a weighted edge as a *link* or *channel*, which corresponds to a single fibre, wavelength, logical transmission channel, or OC-n signal unit between a pair of nodes. The collection of links in parallel on the edge is called the *span*, which corresponds to the entire set of unit capacity links, i.e., the whole system of transmission equipment including the optical fibre bundles, cables, and ducts between a pair of nodes. The weight of an edge in the graph, and the number of links it consists of, is called the span's *capacity*.

The term span as used herein has its origin in the transmission networking community where it referred to a grouping of physical layer carrier signals between adjacent cross-connecting nodes that can undergo a common-cause failure. Bhandari [4] explains that "...spans are the set of physical transmission fibres / cables in the physical facility graph. Links of the logical connectivity graph are built from spans. A given span can thus be common to a number of links." We further define a span as constituting the set of all physical working and spare channels that terminate on adjacent cross-connecting nodes and share a common exposure to a single physical cut of their infrastructure, such as a duct or cable. Each working or spare capacity unit on a span is destined to fail together if the corresponding physical span fails[1].

A sequence of $|W|$ adjacent edges is called a *walk* of *length* $|W|$ and is denoted as the ordered set $W = \left\{e_1, e_2, e_3, \ldots, e_{|W|}\right\}$ where e_i are the edges in the walk in the order in which they are crossed. A walk can also be denoted as the ordered set $W = \left\{(v_1, v_2), (v_2, v_3), \ldots, \left(v_{|W|}, v_{|W|+1}\right)\right\}$, or alternatively, as $W = \left\{v_1, v_2, v_3, \ldots, v_{|W|}, v_{|W|+1}\right\}$ where v_i are the vertices in the walk in the order in which they are crossed. Note that there is exactly one more vertex in a walk than there are edges in it. The first vertex in the walk is called its *origin* and the last vertex is called its *destination* or *terminus.* A walk with distinct edges (i.e., there are no duplicate edges) is a *trail*, and a trail with distinct vertices (except possibly its origin and terminus) is a *path* or a *chain.* Two paths (or walks or trails) are *edge-disjoint* if they have no edge in common, and are *vertex-disjoint* (or simply *disjoint*) if they have no vertex in common. Where we wish to consider (or determine) multiple disjoint paths between a pair of origin and destination vertices, the origin and destination are often disregarded in establishing the paths' disjointedness.

---

[1] Notwithstanding the specific meaning of *span* here, readers are advised that some members of the industry often also use the more generic term *link* in this same context. The intended meaning of *link* as either a service-layer or physical-layer entity should be construed appropriately in each case.

In transport networking, a *path* refers to a specific end-to-end concatenation of cross-connected links, upon which a unit transmission signal would be borne. In this context, a path is by definition composed of distinct links since (at least in transport networking) a link cannot be cross connected to more than a single other link on either end. An end-to-end concatenation of spans is called a *route.* Thus, every path follows a route, and several paths could share a route or portions of their routes. We do not make a distinction between a route that is composed of distinct spans (i.e., the route does not cross the same span twice) and one that is not, although in most cases, the former is typical in transport networking. Consequently, it is possible for two (or more) of a path's constituent links to cross the same span, which is the case in a collapsed ring [99].

A closed trail is a *tour*, while a closed path is a *cycle.* Tours and cycles are the basic graph structure involved in ring and *p*-cycle restoration, and in transport networking it is common for both tours and cycles to be referred to as cycles. We will henceforth refer to either tours or cycles as cycles. As defined in this context, a cycle that closes on itself only once (i.e., it does not contain a "figure-eight") is called a *simple cycle.* A cycle crossing all edges in a graph is an *Eulerian cycle*, and a graph containing at least one is called an *Eulerian graph.* A cycle traversing all vertices of a graph is a *Hamiltonian cycle*, and a graph containing at least one is called a *Hamiltonian graph.*

Two vertices in a graph are *connected* if there exists at least one path between them, and a graph G is *connected* if every pair of vertices in G is connected. If two vertices of a graph are not connected, they are said to be *disconnected.* A graph $G' = (V', E')$ is called a *component* of graph $G = (V, E)$ if $G' \subseteq G$ and there is no vertex or edge that is in both G' and $\bar{G}'$. In other words, G' is a component of G if every vertex in G' is connected to all of the other vertices in G' and disconnected from all of the vertices in $\bar{G}'$. Components are by definition, disconnected subgraphs of a graph. A graph is *two-connected* if there are at least two edge-disjoint paths between every pair of its vertices, and is *bi-connected* if there are at least two vertex-disjoint paths between every pair of its vertices. A bi-connected graph is also a two-connected graph, but the reverse is not necessarily so. In transport networks, two-connectivity is strictly required for survivability to any single span (edge) failure, while bi-connectivity is required for survivability to any single node (vertex) failure. Refer to Figure 2.3 for examples of such graphs. While it is difficult and time-consuming for an algorithm to identify a graph as being two-connected, bi-connected, or neither [89], it is usually trivial to do so visually by looking for *articulation points*. In transport networking terminology, a *stub node* is a node whose degree is one, a *bridge node* is a node whose removal would partition the network graph into two disconnected components, and a *bridge node* is a node whose removal would disconnect the graph. A bi-connected graph will have no such articulation points, while a two-connected (but not bi-connected) graph will have no stub nodes or bridge spans, but may have a bridge node. Bi-connected graphs are also referred to as *closed* graphs.

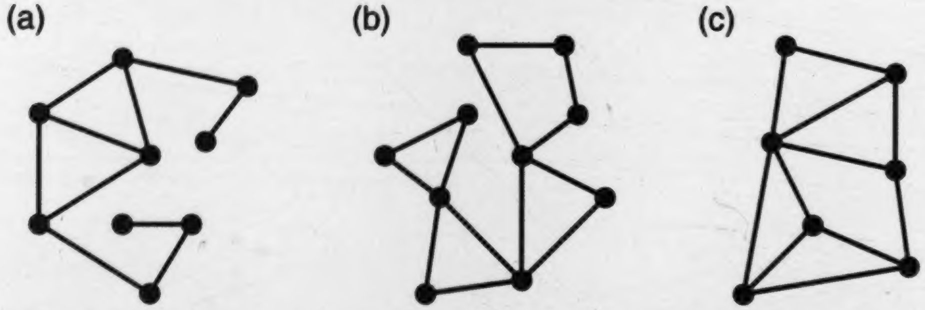


Figure 2.3 – Examples of (a) connected, (b) two-connected, and (c) bi-connected graphs.

Any graph that contains no cycles is called an *acyclic graph*. A connected acyclic graph is called a *tree*. A disconnected acyclic graph is a *forest*, and hence is composed of components who are themselves each a tree. An acyclic subgraph $G' \subseteq G$ that contains every vertex in G is called a *spanning tree*, and a set of disconnected subgraphs $G_1 \subseteq G$, $G_2 \subseteq G$, $G_3 \subseteq G$, ... that as a whole contain every vertex in G is called a *spanning forest*. Examples are shown in Figure 2.4.

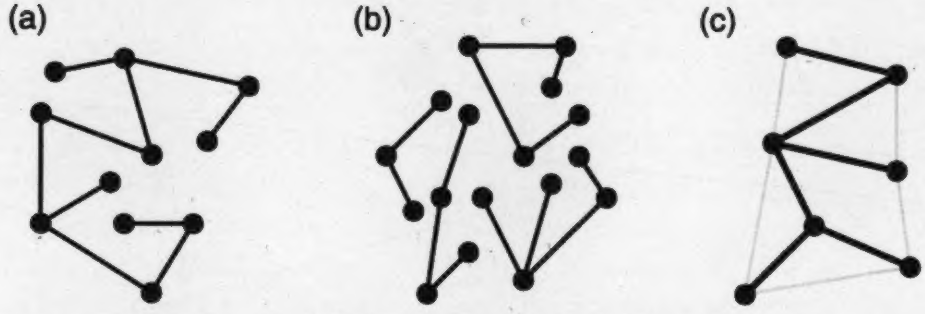


Figure 2.4 – Examples of (a) a tree, (b) a forest, and (c) a spanning tree (in bold).

2.3 OPERATIONS RESEARCH TERMINOLOGY AND NOTATION

Operations research is the science of study, design, analysis, and operation of decision-making problems and systems with limited resources, with the goal of optimizing a system's performance [119]. *Mathematical programming* is the study of a class of problems that seek to either maximize or minimize a mathematical equation (the *objective function*) relating to one or numerous decision variables, subject to specified variable constraint equations (*constraints*). Such problems are called *mathematical programming problems* and can always be reduced to the following algebraic form:

$$\text{Minimize} \quad f(x_1, x_2, x_3, x_4, \ldots) \tag{2.1}$$

$$\text{Subject to:} \quad f_i(x_1, x_2, x_3, x_4, \ldots) \geq c_i \qquad \forall i \in C \tag{2.2}$$

Here, $x_1, x_2, x_3, x_4, \ldots$ are decision variables and equation (2.1) is the objective function. The equations in (2.2) are the constraints, while C is the set of such constraints and all c_i are constants.

Mathematical programming problems can be divided into two main types. A *non-linear programming problem* (NLP) is a mathematical programming problem whose objective function and/or at least one constraint is in the form of a non-linear equation, while a *linear programming problem* (LP) is a mathematical programming problem whose objective function and constraints can all be expressed as linear equations. All of the problems dealt with in this thesis are LPs, and so we will not consider NLPs further. LPs can always be reduced to the following algebraic form:

$$\text{Minimize} \quad \sum_{\forall i} a_i \cdot x_i \tag{2.3}$$

$$\text{Subject to:} \quad \sum_{\forall i} b_{i,j} \cdot x_i \geq c_j \qquad \forall j \in C \tag{2.4}$$

Here, a_i and $b_{i,j}$ are all constants, and other notation is as already defined above.

Bounds are a special class of constraints often used in LPs, and take the following form:

$$x_i \leq d_i \qquad \forall i \tag{2.5}$$

where the inequality in equation (2.5) can be any of the set $\{\leq,<,=,>,\geq\}$.

An *integer linear programming problem* (ILP) is an LP where at least one of the variables is restricted to integer values. A *mixed integer programming problem* (MIP) is an ILP where at least one of the variables is continuous (i.e., not restricted to integer values), while a *pure integer programming problem* is an ILP where all of the variables are required to be integer valued. ILPs can always be reduced to the same algebraic form as LPs with one additional set of "constraints" for each variable that is restricted to integer values:

$$x_i \in \{\text{integer numbers}\} \tag{2.6}$$

Unless otherwise specified, problems discussed in this thesis are ILPs, and in most cases they are pure ILPs, although for reasons to be discussed later, we typically solve them as MIPs. In cases where we solve a pure ILP as an MIP, we *relax* (disregard) the integrality constraint on one or more variables. In some cases, all integrality constraints in an ILP are relaxed, creating the *LP-relaxation* (or LP version) of the ILP. The main reason one might solve an LP-relaxation of an ILP is that LPs are much easier to solve than ILPs, and the solution to the LP-relaxation is known to be a lower bound on a minimization ILP and the upper bound on a maximization ILP [119]. Variables that are restricted to integer values are called *integer variables*. *Binary variables* are a special class of integer variable whose values are restricted to either 1 or 0. Our experience shows that ILPs with binary variables tend to be more difficult to solve than those without them.

2.3.1 SOLVING LINEAR PROGRAMMING PROBLEMS

The *simplex method* [20], [119], is a simple and efficient algorithm for solving linear programming problems developed in 1947 by George Dantzig, then a Mathematical Advisor at the U.S. Defense Department. The simplex method involves a series of matrix operations on a standard matrix form of the LP, and while it is possible to perform by hand on relatively small LPs with just a few variables and constraints, doing so on the size of problem addressed in this thesis is impossible. An interested reader can refer to [119] for a thorough description of the simplex method, however, for the purposes of this thesis, we only note that such a method exists.

In solving the LPs we deal with within this thesis, we use third party software. AMPL [35] is a commercial software package and modeling language used to efficiently describe mathematical programming formulations in a simple algebraic form. Using AMPL, we first develop a basic model that describes the LP in a general sense. Then we provide the AMPL software with the LP model and data specific to the network or test case to generate a description of the specific LP we wish to solve in a standard file format. Finally, we use CPLEX [61], a very sophisticated LP solver to find the optimal solution to the LP.

2.3.2 BRANCH AND BOUND HEURISTIC FOR SOLVING ILPS

The simplex method can be used for solving non-integer LPs only; it is not capable of solving ILPs, which are, in general, much more difficult to solve [119]. Since this thesis will deal with ILPs so much, a brief discussion of the *branch-and-bound heuristic* for solving such problems is warranted. The first step in the branch-and-bound heuristic is to solve the LP-relaxation of the ILP. If the solution is one in which all variables that must take on integer values do so, then this is also the optimal solution of the ILP. However, for most ILPs, the values of at least several integer variables will not be integer in the solution of its LP-relaxation. Referring to the *branch-and-bound tree* in Figure 2.5, consider the case where we are solving a minimization MIP where variables x_1 and x_2 are integer variables (the other variables, not shown, are not required to be integer). Also consider that both of those variables have existing upper bounds of 10 and lower bounds of 0 (they must take on values between 0 and 10, inclusive); it is not only common, but also standard that decision variables in LPs have upper and lower bounds. Solving the LP-relaxation of the MIP results in $x_1 = 3.4$ and $x_2 = 6.8$, with an objective function value of 151. This sub-problem solution becomes node **1** at the apex of the branch-and-bound tree. Noting that neither integer variable x_1 nor x_2 have integer values, we arbitrarily select x_1 to "branch" on, and create two new sub-problems (at nodes **2** and **3**) by introducing new bounds for that variable. Since x_1 is an integer variable, we know that it cannot possibly have a value that is *between* 3 and 4, yet at node **1**, its value is 3.4. So if we add a bound to restrict x_1 to be 3 or less in one new sub-problem and 4 or greater in the other, then we know that the optimal solution of the MIP must be found in one of the two new sub-problems. This has effectively partitioned the *feasible region* (the space of possible values that will result in all constraints and bounds being satisfied, but will not necessarily result in an optimal solution) into two parts. We can also note that the original non-integer solution found at node **1** cannot possibly re-occur.

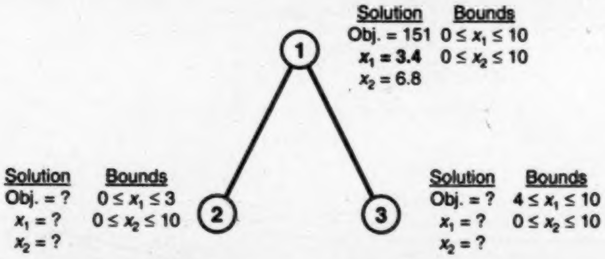


Figure 2.5 – Illustrating the first steps of the branch-and-bound heuristic.

Now as shown in Figure 2.6, we arbitrarily select node **2**, and solve an LP-relaxation of that sub-problem, resulting in a solution of $x_1 = 2.4$ and $x_2 = 7.6$, with an objective function value of 156. Again, we arbitrarily chose one of the two integer variables with fractional values and branch on it, in this case, x_2. We add one new bound restricting x_2 to 7 or less to create the sub-problem at node **4**, and another restricting x_2 to 8 or greater to create the sub-problem at node **5** (and in both cases, the bounds added at node **2** remain in effect as well).

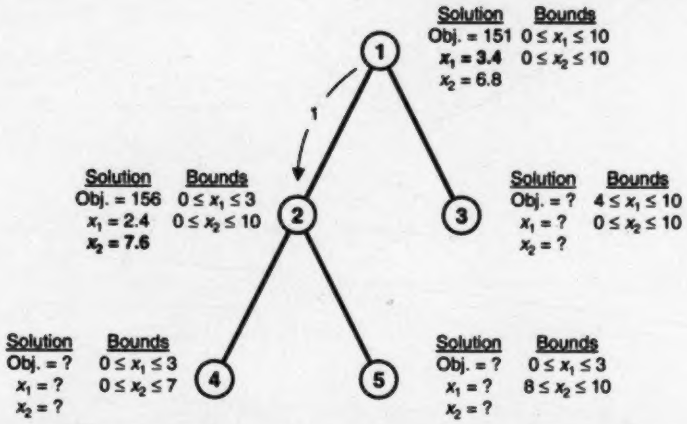

Solution
Bounds
Obj. = 151
$0 \le x_1 \le 10$
$x_1 = 3.4$
$0 \le x_2 \le 10$
$x_2 = 6.8$
1
1
Solution
Bounds
Obj. = 156
$0 \le x_1 \le 3$
$x_1 = 2.4$
$0 \le x_2 \le 10$
$x_2 = 7.6$
2
3
Solution
Bounds
Obj. = ?
$4 \le x_1 \le 10$
$x_1 = ?$
$0 \le x_2 \le 10$
$x_2 = ?$
Solution
Bounds
Obj. = ?
$0 \le x_1 \le 3$
$x_1 = ?$
$0 \le x_2 \le 7$
$x_2 = ?$
4
5
Solution
Bounds
Obj. = ?
$0 \le x_1 \le 3$
$x_1 = ?$
$8 \le x_2 \le 10$
$x_2 = ?$

Figure 2.6 – Illustration of branch-and-bound heuristic with node 2 solved.

Using the depth-first, or last-in-first-out rule (LIFO)[2], we choose to solve one of the most recently created sub-problems (we arbitrarily choose node **4**) rather than go back up to node **3**. Solving the LP-relaxation of the sub-problem at node **4**, we obtain a solution of $x_1 = 3.0$ (integer) and $x_2 = 6.0$ (integer), with an objective function value of 169, as shown in Figure 2.7. Here, both integer variables actually take on values that are integer, and so this becomes our first *candidate solution.* Since no integer variables have fractional values, then we do not perform a branch-and-bound operation at node **4**, and that branch of the tree can be considered *fathomed.* So we now solve the LP-relaxation of the sub-problem at node **5**, obtaining the solution of $x_1 =$ 2.9 and $x_2 = 8.0$, with an objective function value of 162. Since the value of x_2 is integer, we now branch on x_1 and create the two sub-problems shown at nodes **6** and **7**, followed by a branch on x_2 at node **6** to create the two sub-problems at nodes **8** and **9**.

---

[2] We could have easily used the breadth-first, or first-in-first-out (FIFO) rule instead, and would then have solved the sub-problem at node **3** instead. There are other more complicated node-selection strategies as well.

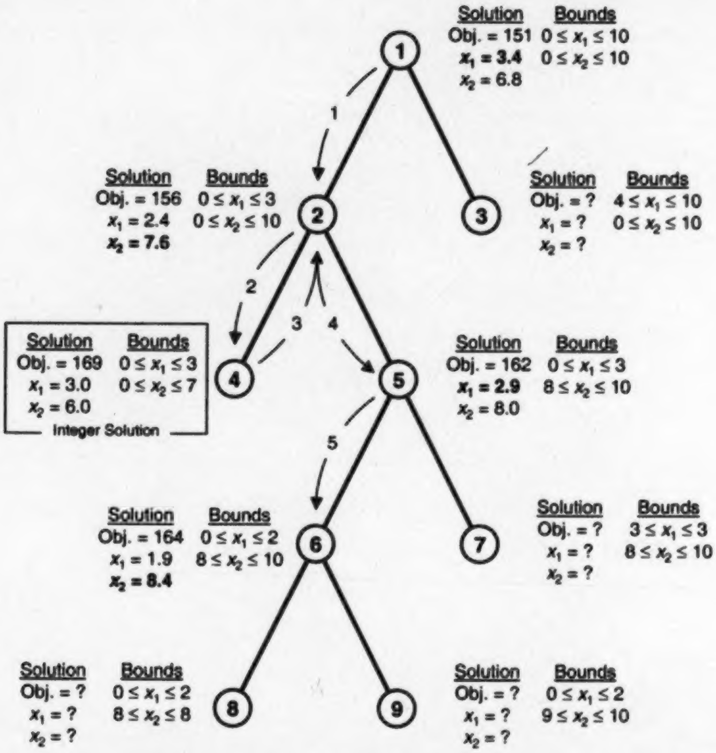

Solution
Obj. = 151
x_1 = 3.4
x_2 = 6.8
Bounds
0 ≤ x_1 ≤ 10
0 ≤ x_2 ≤ 10
1
Solution
Obj. = 156
x_1 = 2.4
x_2 = 7.6
Bounds
0 ≤ x_1 ≤ 3
0 ≤ x_2 ≤ 10
2
3
Solution
Obj. = ?
x_1 = ?
x_2 = ?
Bounds
4 ≤ x_1 ≤ 10
0 ≤ x_2 ≤ 10
2
3
4
Solution
Obj. = 169
x_1 = 3.0
x_2 = 6.0
Bounds
0 ≤ x_1 ≤ 3
0 ≤ x_2 ≤ 7
Integer Solution
4
5
Solution
Obj. = 162
x_1 = 2.9
x_2 = 8.0
Bounds
0 ≤ x_1 ≤ 3
8 ≤ x_2 ≤ 10
5
Solution
Obj. = 164
x_1 = 1.9
x_2 = 8.4
Bounds
0 ≤ x_1 ≤ 2
8 ≤ x_2 ≤ 10
6
7
Solution
Obj. = ?
x_1 = ?
x_2 = ?
Bounds
3 ≤ x_1 ≤ 3
8 ≤ x_2 ≤ 10
Solution
Obj. = ?
x_1 = ?
x_2 = ?
Bounds
0 ≤ x_1 ≤ 2
8 ≤ x_2 ≤ 8
8
9
Solution
Obj. = ?
x_1 = ?
x_2 = ?
Bounds
0 ≤ x_1 ≤ 2
9 ≤ x_2 ≤ 10

Figure 2.7 – Illustration of branch-and-bound heuristic with 5 nodes solved.

Solving the LP-relaxation of the sub-problem at node **8** (Figure 2.8) gives us another integer solution of $x_1 = 2.0$ and $x_2 = 8.0$, with an objective function value of 166, slightly better than the previous candidate solution at node **4** (remember that we're trying to minimize the objective function). Therefore, we discard the candidate solution at node **4**, and the solution at node **8** becomes our new candidate solution (and the branch is fathomed).

Next, we solve the LP-relaxation of the sub-problem at node **9** and find that this sub-problem is *infeasible*, or in other words, it is impossible to find a solution to this sub-problem where all constraints and bounds are satisfied. This branch can also be considered fathomed since further branching from this node cannot possibly yield a valid solution (an infeasible problem cannot possibly be made feasible by adding more bounds or constraints).

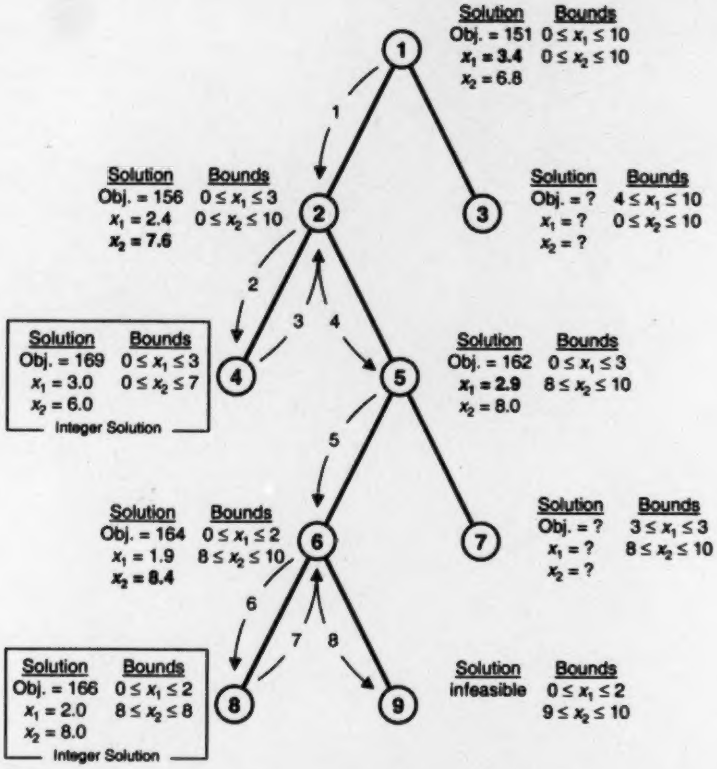

1
Solution
Bounds
Obj. = 151
$x_1 = 3.4$
$x_2 = 6.8$
$0 \le x_1 \le 10$
$0 \le x_2 \le 10$
2
Solution
Bounds
Obj. = 156
$x_1 = 2.4$
$x_2 = 7.6$
$0 \le x_1 \le 3$
$0 \le x_2 \le 10$
3
Solution
Bounds
Obj. = ?
$x_1 = ?$
$x_2 = ?$
$4 \le x_1 \le 10$
$0 \le x_2 \le 10$
4
Solution
Bounds
Obj. = 169
$x_1 = 3.0$
$x_2 = 6.0$
$0 \le x_1 \le 3$
$0 \le x_2 \le 7$
Integer Solution
5
Solution
Bounds
Obj. = 162
$x_1 = 2.9$
$x_2 = 8.0$
$0 \le x_1 \le 3$
$8 \le x_2 \le 10$
6
Solution
Bounds
Obj. = 164
$x_1 = 1.9$
$x_2 = 8.4$
$0 \le x_1 \le 2$
$8 \le x_2 \le 10$
7
Solution
Bounds
Obj. = ?
$x_1 = ?$
$x_2 = ?$
$3 \le x_1 \le 3$
$8 \le x_2 \le 10$
8
Solution
Bounds
Obj. = 166
$x_1 = 2.0$
$x_2 = 8.0$
$0 \le x_1 \le 2$
$8 \le x_2 \le 8$
Integer Solution
9
Solution
Bounds
infeasible
$0 \le x_1 \le 2$
$9 \le x_2 \le 10$
1
2
3
4
5
6
7
8

Figure 2.8 – Illustration of branch-and-bound heuristic with 7 nodes solved.

Backing up the branch-and-bound tree in Figure 2.9, we next visit the sub-problem at node **7** and obtain a solution of $x_1 = 3.0$ and $x_2 = 9.0$, with an objective function value of 168. This is another integer solution, but since the objective function is greater (worse) than our current candidate solution, we discard it and this branch is also fathomed. Now we back up to the last unsolved sub-problem in the branch-and-bound tree (node **3**), and solve its LP-relaxation to obtain a solution of $x_1 = 4.0$ and $x_2 = 7.4$, with an objective function value of 167. Although this solution has an integer variable with a fractional value, we can observe that the objective function value of 167 is worse than the objective function value of the current candidate solution. Branching on x_2 at this node can only ever result in subsequent sub-problems with objective function values no better than 167 (we would need to be able to *remove* some bounds to improve on the objective function), so this final node is also fathomed[3]. Since there remain no unsolved sub-problems, then the current best candidate solution we obtained at node **8** is the optimal integer solution to the MIP.

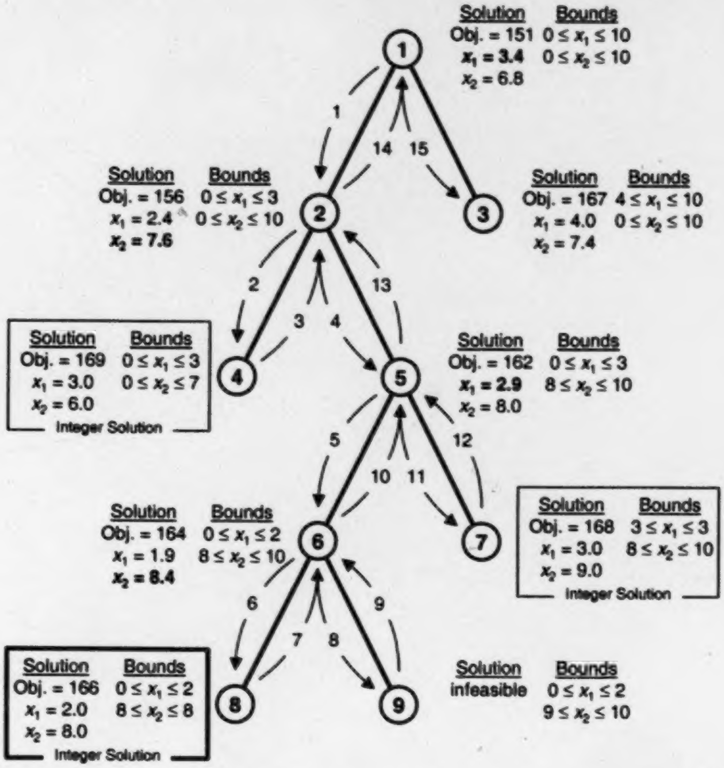

Solution Bounds
Obj. = 151 $0 \le x_1 \le 10$
$x_1 = 3.4$ $0 \le x_2 \le 10$
$x_2 = 6.8$
1
1
14 15
Solution Bounds
Obj. = 156 $0 \le x_1 \le 3$
$x_1 = 2.4$ $0 \le x_2 \le 10$
$x_2 = 7.6$
2
3
Solution Bounds
Obj. = 167 $4 \le x_1 \le 10$
$x_1 = 4.0$ $0 \le x_2 \le 10$
$x_2 = 7.4$
2
13
3 4
Solution Bounds
Obj. = 169 $0 \le x_1 \le 3$
$x_1 = 3.0$ $0 \le x_2 \le 7$
$x_2 = 6.0$
Integer Solution
4
5
Solution Bounds
Obj. = 162 $0 \le x_1 \le 3$
$x_1 = 2.9$ $8 \le x_2 \le 10$
$x_2 = 8.0$
5
12
10 11
Solution Bounds
Obj. = 164 $0 \le x_1 \le 2$
$x_1 = 1.9$ $8 \le x_2 \le 10$
$x_2 = 8.4$
6
7
Solution Bounds
Obj. = 168 $3 \le x_1 \le 3$
$x_1 = 3.0$ $8 \le x_2 \le 10$
$x_2 = 9.0$
Integer Solution
6
9
7 8
Solution Bounds
Obj. = 166 $0 \le x_1 \le 2$
$x_1 = 2.0$ $8 \le x_2 \le 8$
$x_2 = 8.0$
Integer Solution
8
9
Solution Bounds
infeasible $0 \le x_1 \le 2$
$9 \le x_2 \le 10$

Figure 2.9 – Illustration of branch-and-bound heuristic with all nodes solved.

---

[3] If the objective function of this sub-problem had been less (better) than 166 (the objective value of the current best candidate solution), then we would continue to explore the branch-and-bound tree from this node, as appropriate.

CHAPTER 3
OPTICAL TRANSPORT NETWORKS

3.1 INTRODUCTION TO TRANSPORT NETWORKS

A *transport network* (also called a *backbone network*) is a network that provides bulk carriage for a variety of communication services and types. The different services are multiplexed together and routed from origin to destination over a common infrastructure of logical multi-channel point-to-point transmission systems. The original transport network was the *public switched telephone network* (PSTN) of twisted pair copper wires connected point-to-point at operator-controlled switching centres. Modern transport networks generally consist of fibre optic cables connected to nodal switching devices such as optical cross-connect switches (OXCs) and add-drop multiplexers (ADMs). While the PSTN was designed to carry voice communications only, modern transport networks carry a variety of services including voice, data, and video. The common infrastructure of the underlying transport network is used to create virtual trunks and circuits to form numerous logical sub-networks, which each carry one of more of the various telecommunication services. Each such logical or virtual sub-network can be operated separately as if they had their own dedicated transmission systems, but in reality they can be thought of as various *service layers* within the underlying transport network. Service layers can be found in a variety of different stacking arrangements, as shown in Figure 3.1, and each acts as a transport network for the next higher layer. One model would see delay-insensitive IP packet data [17] served over an ATM layer [34] that carries its own delay-sensitive packet data payloads, and the ATM layer could in turn be served by a WDM layer [68], which is carried directly over fibre. Other stacking models may also utilize PDF [6], SONET/SDH [6], GMPLS [21], or other layers. Detailed description of the service layers listed above is outside of the scope of this thesis. However, because most of the work discussed herein applies most directly to SONET and WDM, brief descriptions of those systems are provided in Sections 3.2 and 3.3, respectively.

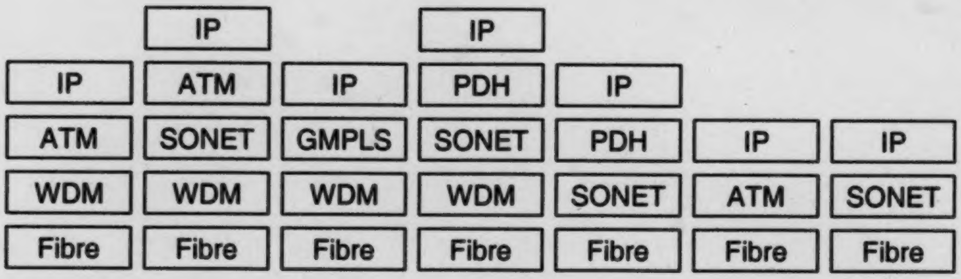


Figure 3.1 – Examples of service layer stacking arrangements in modern transport networks.

3.1.1 TRANSPORT NETWORK PARTITIONING

Transport networks can also be partitioned conceptually into three main tiers based on administrative, political, and/or geographical boundaries, as shown in Figure 3.2. The *access network* connects local customer premises or remote switching centres to nearby *central offices* (COs), which in turn act as hubbing points for the typically small demands originating or terminating at the various remote locations. Access networks often tend to be quite tree-like or otherwise very sparse in topology, and in many residential areas may not necessarily be survivable (bi-connected). *Metro* (*inter-office*) *networks* connect regional COs and other hubbing centres together, and typically span relatively short distances of no more than 50 km across. Metro network costs tend to be dominated by nodal equipment costs (ADMs, OXCs, etc.) because of the relatively short span distances involved; fibre, cable installation, and rights-of-way costs on such a small scale are rather low by comparison. Any traffic entering/leaving a metro network does so at peering points called *points of presence* (POPs), which are co-located on a *long-haul network*, which connects numerous metro networks within a large region, or even on a national or international scale. Long-haul networks are also called *inter-exchange networks* because they carry traffic between various metro networks. Since aggregate span distances in even a small long-haul network can easily reach 1000s of kilometres, their costs are dominated by fibre, rights-of-way, and other distance-related costs (amplifiers, regenerators, etc.). Any traffic crossing from one long-haul network into another does so at peering points called *network access points* (NAPs), which are co-located nodes on both networks.

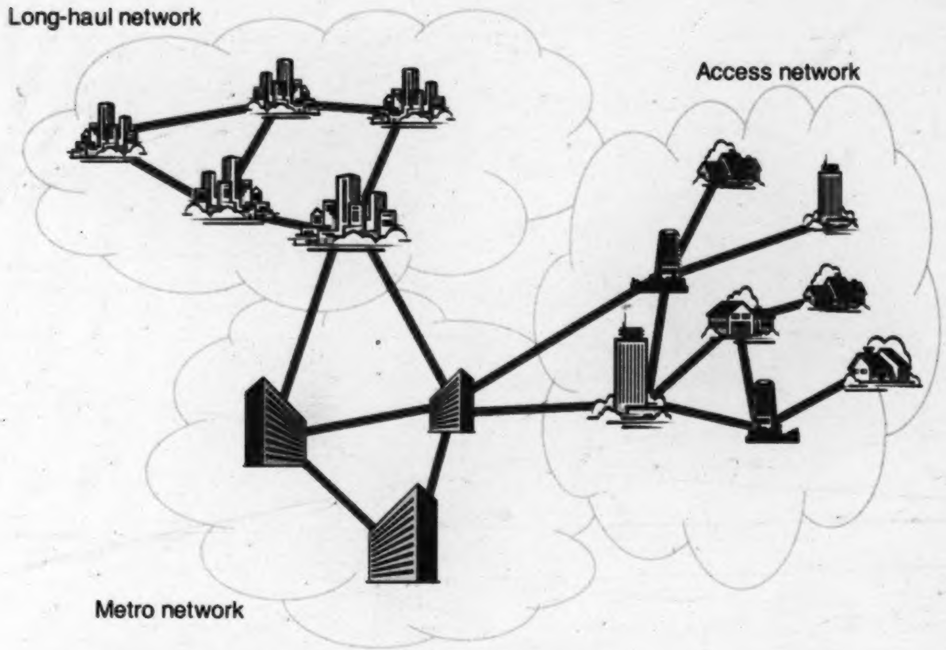


Figure 3.2 – Partitioned view of a modern transport network.

3.1.2 OSI TRANSPORT LAYER

Before going any further, we need to take care to distinguish between the transport network as defined herein and the transport layer of the well known Open Systems Interconnection (OSI) model. The OSI model is "a layered framework for the design of network systems that allows for communications across all types of computer systems" [34]. It was developed by the International Standards Organization to assist in and promote the development of flexible but robust interoperable network protocols. The OSI model, shown in Figure 3.3, consists of seven ordered layers: (1) physical, (2) data link, (3) network, (4) transport, (5) session, (6) presentation, and (7) application. Each layer is intended to characterize a set of networking functions, such that data and network information can be passed up or down through the adjacent layers via layer interfaces that define what services one layer must provide to the layer above it in the stack. This allows implementations in one layer to be modified as desired without requiring the adjacent layers to be changed as well. The lower four layers in the OSI model (the physical, data link, network, and transport layers) deal with the physical aspects of moving data, and incorporate electrical specifications, physical connection standards, reliability issues, and synchronization/timing, among other specifications. It is these layers that encompass the transport network as we define it. For more information on the OSI model, refer to [34].

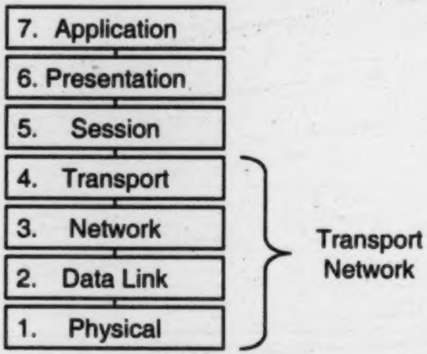


Figure 3.3 – The OSI model and how it relates to the transport network.

3.1.3 TRANSPORT NETWORKING TECHNOLOGIES

Transport networks make use of a variety of different components in their operation. A detailed discussion of most of them is beyond the scope of this thesis, however, we will briefly describe several key components to facilitate later discussions. There are two main types of nodal devices we will refer to frequently in this thesis: an *Optical Cross-Connect* (OXC) and an *Optical Add/Drop Multiplexer* (OADM) [111].

The ability to cross connect channels from one line to another is a key function required in modern optical communication systems. An OXC is a line-terminating device with interfaces for hundreds or thousands of input/output lines, as well as local add and drop ports, as illustrated conceptually in Figure 3.4. At the centre of an OXC is a cross-connecting fabric with the capability to connect any input with any output. Most OXCs available today employ a hybrid cross-connection approach, where optical signals are converted into electrical signals and cross-connection is done in the electrical domain. While wavelength conversion is inherent in hybrid OXCs since the re-generated optical signal on the output side can be applied to any wavelength as needed for routing or other considerations, it is not in all-optical OXCs (although some of the latter do contain integrated wavelength converters).

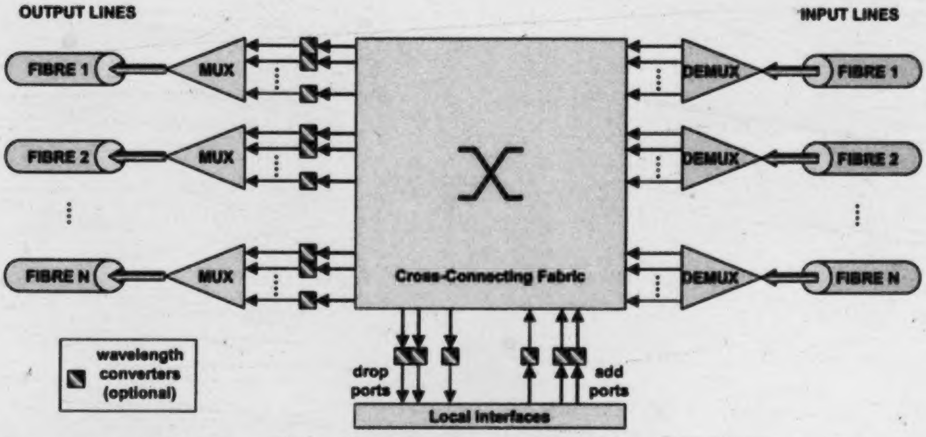

OUTPUT LINES
INPUT LINES
FIBRE 1
FIBRE 2
FIBRE N
MUX
DEMUX
Cross-Connecting Fabric
wavelength converters (optional)
drop ports
add ports
Local interfaces

Figure 3.4 – A simple optical cross-connect (OXC).

All-optical OXCs are expected to dominate in the future, although they are currently much more complicated and cannot yet approach the size of the hybrid versions. Micro-electro-mechanical switches (MEMS) are perhaps the most promising all-optical switching technology, and those with as many as a thousand inputs and outputs are currently in the experimental phase [68]. One MEMS all-optical switch now available is Lucent Technologies' WaveStar™ LambdaRouter Optical Cross-Connect [79], in which two sets of 256 tilting mirrors are arranged on two 1-inch square chips. The LambdaRouter is shown in Figure 3.5.

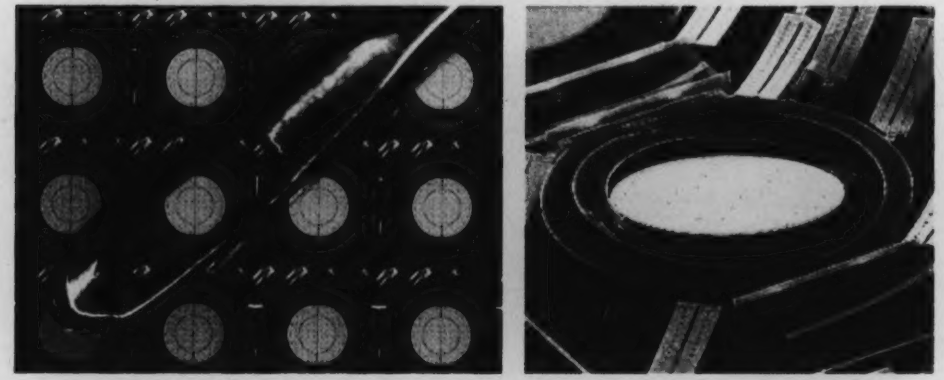


Figure 3.5 – WaveStar™ LambdaRouter Optical Cross-Connect Switch: with a standard sewing needle to show scale (left) and a close-up of one mirror partially tilted (right), copyright 2004, Lucent Technologies.

An ADM is a line-terminating device with interfaces for only two main lines (typically referred to as East and West lines), as well as local add and drop ports, as shown in Figure 3.6. The add and drop ports allow tributary signals to be added to or dropped from (enter or exit) the main line signals passing through the ADM via the East and West lines. One common use of ADMs is in survivable rings (see Section 4.3) where half of the capacity in and out of each line is used to carry working capacity, while the other half is reserved for spare capacity. An *optical* ADM (OADM), also called a *wavelength* ADM (WADM) may also include wavelength converters [68], as shown in the figure.

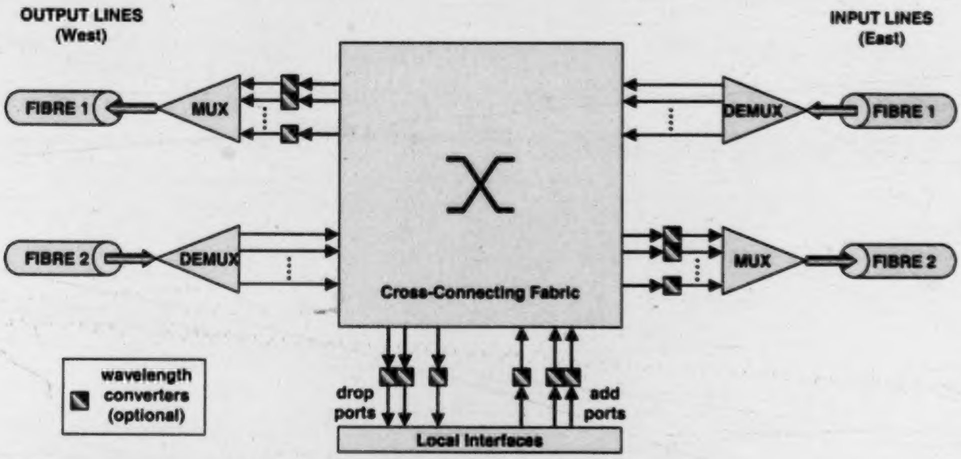

OUTPUT LINES
(West)
INPUT LINES
(East)
FIBRE 1
MUX
DEMUX
FIBRE 1
FIBRE 2
DEMUX
MUX
FIBRE 2
Cross-Connecting Fabric
wavelength converters (optional)
drop ports
add ports
Local Interfaces

Figure 3.6 – A simple optical add/drop multiplexer (OADM).

For more information on the various other optical network enabling technologies, including optical fibre, filters, optical amplifiers, multiplexers and de-multiplexers, wavelength converters, lasers, etc., and detailed discussions of some of the scientific and technological challenges to optical networking, refer to [68] and [96].

3.2 SONET

The most widely known optical-based transport networking standard is the *Synchronous Optical Network* (SONET) standard [6], [48], [96], [110], [111], which traces its roots to the break-up of AT&T in 1984. With the large number of new companies all competing and developing equipment to operate on their own proprietary transmission schemes, it was recognized that some sort of cooperation would be needed to ensure compatibility of their equipment and networks. The American National Standards Institute (ANSI) established the *synchronous transport signal number one* (STS-1) rate of 51.84 Mb/s as a base standard, which led to the publication of a draft SONET standard in 1987. Beginning in the early 1990s, SONET and Europe's related and compatible SDH[4] standard have become the dominant standards for *time division multiplexed* (TDM) transport networks worldwide. Some of the features that make SONET so successful include:

- robustness and high amenability to survivability
- very accurate clock synchronization operations
- ease of multiplexing, de-multiplexing, and traffic grooming
- operations, administration, maintenance, and provisioning (OAM&P) services
- flexibility and support for a variety of payloads

The SONET STS-1 base transport signal format is a frame of duration 125 μs (corresponding to 8000 frames per second) and is a total of 810 bytes in size, structured logically as 90 columns of 9 rows of bytes, as shown in Figure 3.7. The first three columns contain *transport overhead* data, which provides signal framing, pointers, monitoring, and control (and more), and the final 87 columns make up the *synchronous payload envelope* (SPE). The first column of the SPE is used for *path overhead* and is used for end-to-end service monitoring, control, and signalling, while the remainder of the SPE is used to carry the actual payload. The STS-1 SPE is capable of carrying up to 28 DS-1 signals (as illustrated in Figure 3.7) or a single DS-3 signal[5].

---

[4] Like SONET, the *synchronous digital hierarchy* (SDH) is a synchronous *time division multiplexing* (TDM) standard. SDH is the official international standard and is common in Europe and Japan, but SONET is dominant in North America. SDH uses the *synchronous transport module level 1* (STM-1) rate of 155.52 Mb/s as the base rate, which allows three STS-1 signals to be multiplexed into one STM-1. This makes SONET and SDH virtually perfectly compatible.

[5] A DS-1 is the *plesiochronous digital hierarchy* (PDH) signal rate of 1.544 Mb/s, which is equivalent to 24 DS-0 voice channels at 64 kb/s. The DS-3 signal rate is 44.736 Mb/s, which is equivalent to 672 DS-0 voice channels. PDH is an *asynchronous* TDM standard that preceded SONET, and is one of the payloads that can be carried by SONET SPEs. For more information on the various PDH standards, refer to [6].

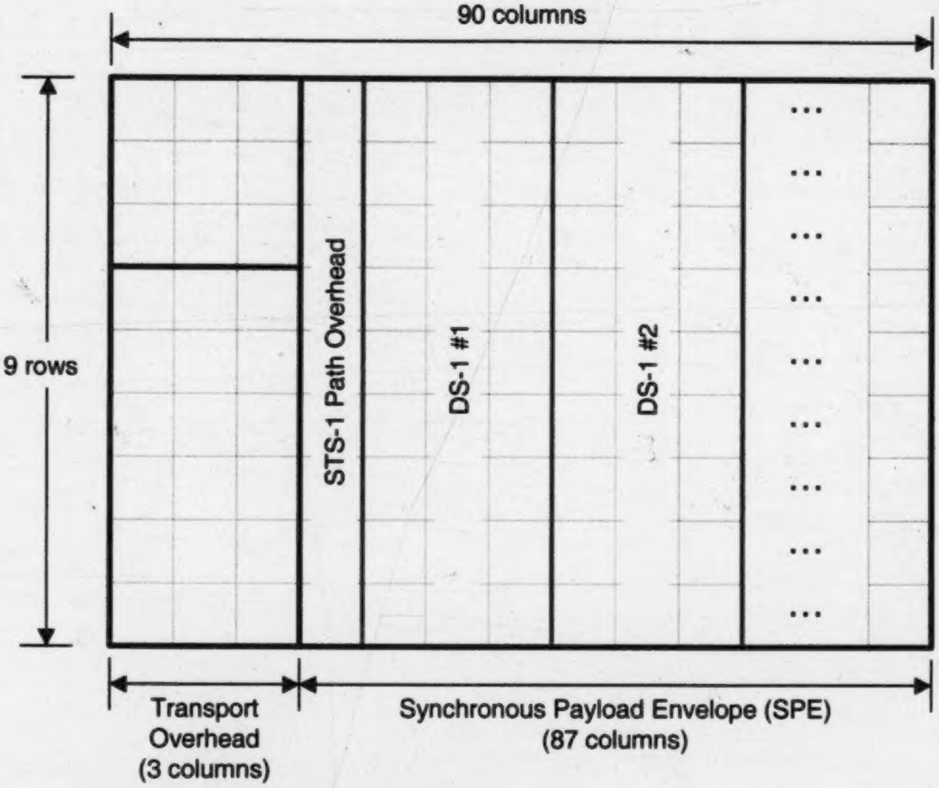

90 columns
9 rows
STS-1 Path Overhead
DS-1 #1
DS-1 #2
...
Transport Overhead (3 columns)
Synchronous Payload Envelope (SPE) (87 columns)

Figure 3.7 – The structure of the SONET STS-1 frame.

SONET data rates higher than STS-1 can be achieved by byte interleaving multiple STS-1s, so all high-rate SONET signals are integer multiples of the STS-1 data rate. These higher rate signals are typically denoted as STS-N where N is the number of multiples of STS-1s interleaved. Typical SONET data rates are listed in Table 3.1. Although SONET applies to optical networks, an STS signal is an electrical signal and (particularly at higher speeds) may only actually exist within SONET equipment. The equivalent rate optical signals are referred to as *optical carrier level N* (OC-N) line rates; these are the optical signals that exist between SONET optical equipment that correspond to the STS-N data rates after electrical-to-optical conversion (so OC-48 is the optical signal corresponding to the STS-48 electrical signal, etc.).

Table 3.1 – Typical SONET data rates.

SONET Signal	Data Rate (Mb/s)
STS-1	51.84
STS-3	155.52
STS-12	622.08
STS-24	1 244.16
STS-48	2 488.32
STS-96	4 976.64
STS-192	9 953.28
STS-768	39 813.12

3.3 WAVELENGTH DIVISION MULTIPLEXING

Wavelength division multiplexing (WDM) is a relatively new transport network technology that is quickly gaining prominence in the industry, primarily due to its ability to greatly increase network transmission capacity at relatively low cost [68], [96], [110], [111]. The basic idea is that a single fibre will transport multiple carrier wavelengths (dozens or even hundreds) simultaneously, with each wavelength capable of carrying its own payload (say, a SONET signal). While any such system is referred to as WDM, a distinction is often made between *coarse* WDM (CWDM) systems where typically two to six wavelengths are applied to a single fibre, and *dense* WDM (DWDM) systems, which use a large number of wavelengths per fibre. In DWDM "dense" refers to the rather tight frequency spacing of wavelengths, while in CWDM, wavelengths are widely separated. DWDM systems with 1000+ wavelengths each operating at 10 Gb/s (OC-192) have been demonstrated. Commercially available systems already operate at the multiple Tb/s rate per fibre [67].

3.3.1 LIGHTPATH ROUTING

Despite its advantages, WDM introduces a new set of problems not seen previously in simpler single-wavelength systems, among them, difficulties with ensuring wavelength continuity and/or wavelength conversion requirements. When a signal is first converted from electrical to optical, it is assigned a specific wavelength, and unless some sort of wavelength conversion can be done en route, then that same wavelength must be available on every span crossed by the signal end-to-end. In most currently deployed networks, this is not a problem because optical signals are converted to electrical at all or most nodal devices along its route. However, it is not so easy in all-optical networks where the signal remains in the optical domain the entire length of its route. Without wavelength conversion at the nodes, it is not uncommon for there to be numerous free wavelengths on a fibre but the specific "colour" required for a given signal is not available – the wavelength mismatch problem.

Over-capacitating the network or placing wavelength converters at the nodes [94] are two solutions to the problem, but doing either can be quite expensive. Using limited wavelength conversion at key nodes is also an option, but then complex *routing and wavelength assignment* (RWA) heuristics [95], [96] are still needed to reduce lightpath blocking. Recent work suggests, however, that *if properly placed*, a small number of wavelength converters in an all-optical network can provide sufficient wavelength conversion capabilities that very little additional capacity is needed to achieve complete routing of all lightpath demands [102].

3.4 NETWORK DEMANDS

In transport networking, the generic term *demand* refers to a working unit of aggregated traffic to be transported between an *origin-destination* (O-D) pair of nodes of the network. The term follows Wu's distinction between traffic itself and the demand units required to transport it [120]. For example, traffic can be thought of as the individual IP packet and/or STS-level tributary flows exchanged between O-D pairs. But demand expresses the aggregate requirement of all traffic types for lightpaths between a given O-D pair. In the context used herein, one unit of demand consumes one working wavelength on each span traversed on the route of the demand between the origin and destination nodes. Throughout this thesis, the amount of traffic corresponding to a unit of demand is equivalent to the amount of traffic that can be carried by a single wavelength, and unless stated otherwise, only integer units of demand are considered (i.e., we will only deal with full lightpaths, not fractional lightpaths). Specific demand models used are detailed in Section 6.2.

CHAPTER 4
NETWORK SURVIVABILITY

As discussed briefly in CHAPTER 1, network outages are very costly, and so it is necessary in today's networks to implement a restoration or recovery mechanism to mitigate the effects of equipment failures. For a number of years, researchers have been developing and studying many types of restoration and protection mechanisms by which networks can recover from failure. In order to set the stage for this thesis, we will first briefly discuss the fundamentals of survivability and describe many of the most common survivability schemes.

4.1 BASIC CONCEPTS

An optical transport network is today required to include an assurance of nearly immediate 100% restoration of all working wavelengths affected by a cable cut (or optical amplifier failure, etc.). Designing for 100% restorability generally means that all of the failed working capacity or working demand units – in this case traffic-bearing lightwave links forming parts of end-to-end lightpaths – can be restored by replacement paths either end-to-end across the network or through detour-like path segments formed between the end-nodes of the failed span itself, through spare capacity distributed throughout the network.

Every span in a mesh-restorable network has a number of working capacity units and a designed-in number of spare capacity units[6] available for such replacement paths. Explicit allocations of spare capacity must then be included in the design of the network. The spare capacity on a span is not, however, for restoration of demands crossing the same span, but is for shared use in restoration routing for *other* span failures. Spare capacity is in every way identical to working capacity but it bears no actual traffic when in the standby state (or traffic it bears is preemptible); it is fully ready for use but is generally not yet cross-connected into any lightpaths until needed to accommodate replacement paths for restoration or protection.

---

[6] In WDM networking, working and spare capacity units are individual WDM carrier wavelengths.

In order to achieve full restorability to any single-span failures, the required replacement paths must be feasible for every such failure scenario within the environment of spare wavelengths surviving after the failure. An obvious aim in designing any survivable mesh network is therefore to assure that all such restoration path-sets are feasible, but within a globally minimized total amount of spare capacity. Also, unlike network design for data communication or call-trunking applications, there is no graceful degradation effect that can be relied upon for resilience (such as cell loss, blocking, or delay increases) if enough spare capacity is not made available through proper solution methods. In a mesh-restorable network the topology, the routing of working flows, and the spare capacity allocation must provide for *complete and exact replacement* of each discrete working capacity unit that may fail. At least in a transport networking context, anything less than an exact matching of each failed wavelength with a restoration path in the spare capacity means abrupt and total outage for all services borne on the affected wavelengths.

4.1.1 DEFINING FAILURE

In the context of this thesis, there are two main types of failures: span failures and node failures. Unless otherwise specified, we consider a span failure to be equivalent to a complete cable cut, which simultaneously severs all transmission capacity (working and spare) on the span. By far the most common cause of cable cuts (approximately 60%) is from work crews accidentally digging them up with backhoes, etc., with another 15% or so caused by vehicle accidents of some kind or human error (say, a worker cutting the wrong cables during maintenance) [18]. FCC statistics estimate that on average metropolitan networks suffer approximately 13 cable cuts annually per 1000 miles of fibre, while long haul networks suffer approximately 3 cuts annually per 1000 miles of fibre. A node failure is equivalent to the simultaneous failure of all spans incident on that node and anecdotal evidence from carriers and operators tells us that, although the consequences of node failures are much more severe than for span failures, they are also much less frequent.

4.1.2 POST-FAILURE

Reversion is the process of returning affected demand flows back to their pre-failure routes from their restoration routes after physical repair of the failed span. With the exception of dedicated 1+1 APS protection, in all cases discussed within this thesis, we are considering networks in which reversion is assumed to occur following a failure and its subsequent repair *before* there is any significant probability of a second failure onset. Mesh-restorable networks can be designed to sustain a second span failure while repair of the first failure is ongoing but the spare capacity penalty can be very high [15] and this is not generally the aim in the practical design of transport networks. It is, however, assumed that in networks where spare capacity is available for *either* restoration or new service provisioning, ongoing provisioning of new service paths during the restored state will have to be cognizant of the spare capacity used by the restoration process and provision new service paths accordingly.

4.2 AUTOMATIC PROTECTION SWITCHING

The simplest and most basic form of network survivability scheme possible is diversely routed *1+1 Automatic Protection Switching* (1+1 APS) as described in [4]. 1+1 APS is an end-to-end path protection arrangement where a primary working channel is duplicated via *head-end bridging* on a dedicated physically diverse backup channel. The receiver monitors both channels, and in the event of failure or signal degradation along the primary channel, it performs a *tail-end transfer* to select the backup channel. Among all the survivability schemes we will discuss, 1+1 APS is the only one that does not allow any degree of sharing of spare capacity resources, and so requires large amounts of spare capacity relative to other methods. In fact, 1+1 APS requires a commitment of *at least* 100% redundancy in terms of the total wavelength-kilometres required for backup routing relative to the number of working wavelengths transported over their shortest paths. Despite its relative capacity inefficiency, 1+1 APS is suitable for use with simple point-to-point terminals for either single-hop or multi-hop channels and can find uses in cases where simple protection protocols are needed, when very fast restoration is crucial, or when capacity efficiency is not the highest concern.

1:1 APS is a related protection mechanism. As in 1+1 APS, each primary channel is paired with an unshared backup channel, but in 1:1 APS, the backup channel does not carry a live copy of the signal. Instead, the backup channel is available to carry a preemptible signal while not needed for protection of the primary channel. Allowing the 1:1 APS protection channels to offer carriage for extra low priority services provides added revenue generation for the network operator. However, 1:1 APS will be slower than 1+1 APS since the receiver must first signal the transmitter that it has detected a failure, and then a head-end bridge must be established to copy the signal onto the protection channel before a tail-end transfer takes place to apply the signal to the appropriate output port.

1:N APS is a more capacity efficient system where the aim is to protect only against isolated channel failures rather than entire cable or system failures, as shown in Figure 4.1. In 1:N APS, a single backup channel is shared amongst multiple primary channels (N of them). In the event of a failure of a working channel, the receiving end first verifies that the protection channel is available, and then signals the other end to establish a head-end bridge of the failed primary channel onto the protection channel before performing a tail-end transfer. *k*:N APS is a more general form of 1:N APS where there are *k* protection channels available instead of just one.

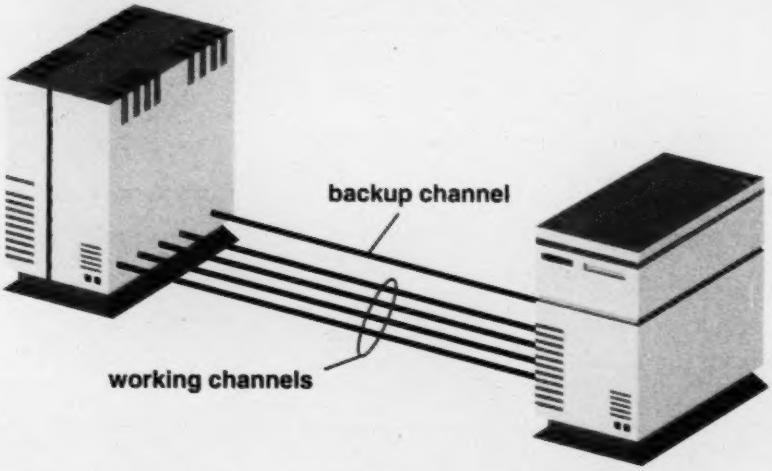


Figure 4.1 – Illustrations of 1:N automatic protection switching.

4.3 SURVIVABLE RING NETWORKS

After APS systems, the next simplest form network survivability is *survivable rings* [86], [48], also called *self-healing rings*. Survivable rings are cyclic structures formed by pre-configuring transmission systems in a ring-like closed loop arrangement using ADM terminal nodal devices, which allow tributary signals to be added to or dropped from (enter or exit) the main line signal within the ring. Half of the transmission capacity between each ADM can be used to carry traffic signals, while the other half is reserved as spare capacity for use in rerouting a failed signal. The cyclic arrangement provides two disjoint routes between any pair of ingress/egress nodes on the ring, and so one such route can act as a redundant backup route for signals on the other route. The self-healing ring protection mechanism restores failures locally (i.e., within the ring itself) and since the structure is completely pre-configured, it is quite fast relative to other restoration mechanism. The two main types of survivable rings are *uni-directional path-switched rings* (UPSRs) and *bi-directional line-switched rings* (BLSRs) [86], [48]. The work described in this thesis will not deal explicitly with ring networks, but for the purposes of providing a more complete discussion and background on network survivability principles in general, a brief description of UPSRs and BLSRs follows.

4.3.1 UNI-DIRECTIONAL PATH-SWITCHED RINGS

The UPSR structure can be viewed simply as a collection of logical 1+1 APS set-ups embedded in a closed loop configuration. In terms of spare capacity requirements, it is no more efficient than using standalone 1+1 APS systems, but network operators like the closed-form nature or survivable rings, and it is better suited to realize transmission capacity economy-of-scale advantages than 1+1 APS. Filling a few high-capacity transmission systems in a ring is often more economical than using several smaller discrete transmission systems in a 1+1 configuration. In addition, the relative simplicity of the ADM terminals used allows STS-type tributary signals to be added or dropped at intermediate nodes. The WDM network equivalent of a UPSR is an *Optical Path Protection Ring* (OPPR) [81].

A UPSR consists of a *working fibre*, which transmits primary signals in one direction (say, clockwise), and a *protection fibre*, which transmits backup copies in the opposite direction (counter-clockwise), as illustrated in Figure 4.2. Note that Figure 4.2 represents the situation where two nodes exchange a bi-directional demand, which is routed clockwise from A to B on the upper-left portion of the ring, and clockwise from B to A on the lower-right portion of the ring. In the event of a span-failure anywhere on the ring, each receiver (at nodes A and B) performs a tail-end transfer to select the backup signal on the protection fibre. Because the backup signal is transmitted in the opposite direction to the primary signal, its survival is assured (in the event of single span failures only). Since each bi-directional pair of signals between any node pair is simultaneously routed in both directions around the ring, then the total capacity required on each fibre (working and protection) on the ring is equal to the sum of the demands routed through it. Protection switching times for a UPSR are generally in the 50 ms range [48], [96].

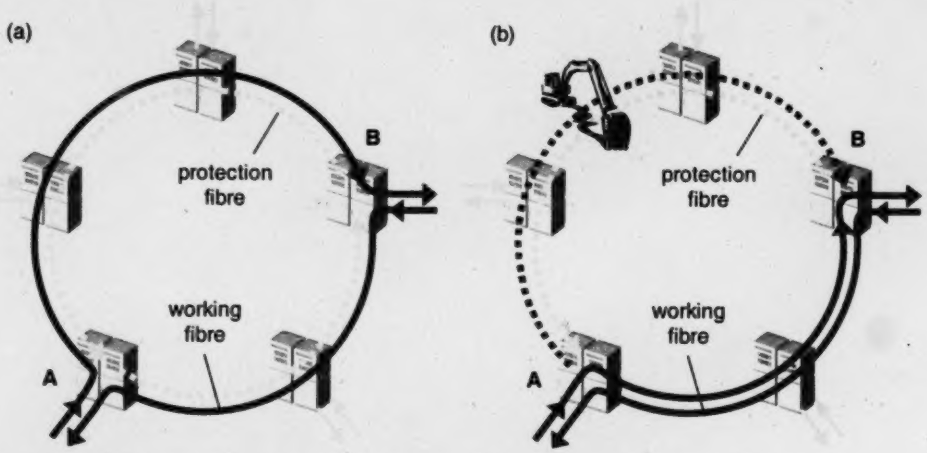


Figure 4.2 – Basic operation of a UPSR (a) before failure, and (b) after failure.

4.3.2 BI-DIRECTIONAL LINE-SWITCHED RINGS

The BLSR structure is the more capacity-efficient ring alternative. In a BLSR, a line-level loop-back mechanism allows protection bandwidth to be shared throughout the entire ring, rather than dedicated for specific backup signals on their protection paths. The WDM network equivalent of a BLSR is an *Optical Shared Protection Ring* (OSPR) [81]. The BLSR has two variants: the *four-fibre BLSR* and the *two-fibre BLSR*. As illustrated in Figure 4.3, adjacent nodes in a four-fibre BLSR are connected by a pair of working fibres and a pair of protection fibres. When a bi-directional demand is exchanged between a pair of nodes on the ring, both directions are routed on the pair of working fibres on one side of the ring (usually the shortest one, though not always), and are not permanently bridged to the protection fibres. In the event of a failure of a span along the working path, the two nodes adjacent to the failed span perform a loop-back operation and insert the failed signals on the protection fibres transmitting in the opposite directions. The two-fibre BLSR operates in a similar fashion, except that adjacent nodes are connected by a single pair of fibres transmitting in each direction, with the channels divided into two halves, a working group and a protection group. The logical configuration is therefore the same as in the four-fibre BLSR.

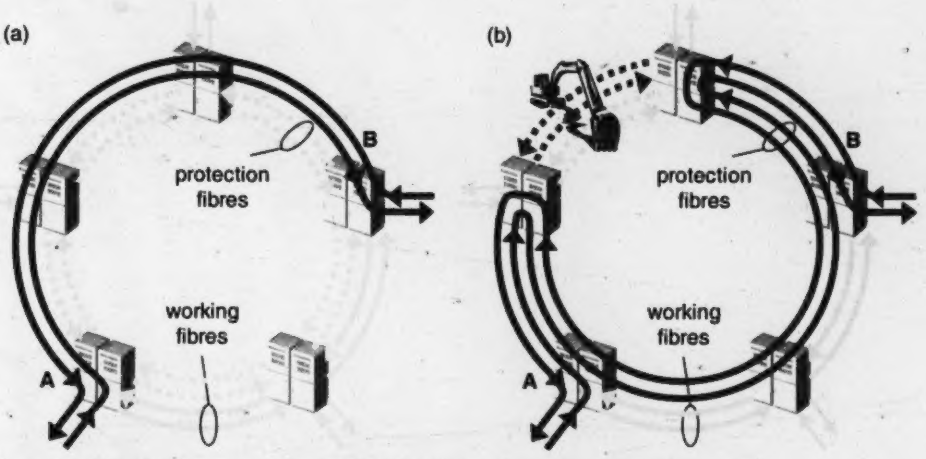


Figure 4.3 – Basic operation of a BLSR (a) before failure, and (b) after failure.

There are two reasons why the BLSR is more capacity efficient than the UPSR. First, since the working signal is routed on only one side of the ring, the channels it occupies can be reused by working signals routed on the other side of the ring, and so a BLSR can carry more demand than a UPSR with the same working capacity. Second, there is no dedicated reservation of protection channels, but rather all working channels share the same capacity on the protection fibres. Nonetheless, the best a BLSR can do is achieve 100% redundancy because the amount of protection capacity around the entire ring must meet or exceed the largest cross-section of working capacity anywhere within the ring. Furthermore, because both nodes adjacent to the failure need to coordinate their signalling and reaction, the operation of a BLSR is somewhat slower than that of a UPSR, and will not necessarily provide 50 ms protection switching times as required by the SONET standards.

4.4 MESH NETWORK SURVIVABILITY

Mesh network survivability refers to a class of mechanism used to ensure network traffic survivability, and includes those methods that allow working routes to follow shortest paths (if desired) and where restoration and protection routes make use of capacity distributed throughout the network rather than in rigidly defined and installed capacity structures (as in rings). Mesh restoration and protection mechanisms allow sharing of spare capacity between multiple service paths and studies show that their spare capacity redundancies can be quite low [25].

We can further categorize survivability schemes by differentiating between *localized* and *end-to-end* restoration and protection. Localized restoration is when replacement paths are established between the end-nodes of the failure itself, whereas end-to-end schemes restore affected demands by constructing replacement paths between their individual origin and destination (O-D) nodes. The latter replacement paths can be completely disjoint from the primary pre-failure service paths, even to the point of re-using idle capacity formerly used (but now released) by pre-failure service paths, depending on the specific scheme implemented. End-to-end survivability schemes are typically able to make more efficient use of network resources, and so tend to require less spare capacity than localized survivability schemes [25].

While ring networks are preferred for their simpler and fast protection mechanism and often find uses in metropolitan and small-scale networks, mesh networks are preferred for their higher bandwidth efficiency and are a good choice in long-distance networks. In addition, while optimally designing ring-based networks has proven to be an extremely difficult problem [86], several essentially exact and complete theories for design, along with relatively simple operational concepts, are now well developed for mesh-based restoration [25], [51], [54], [60], [63], [121]. Mesh-based transport network design will be the focus of the research discussed in this thesis.

4.4.1 SPAN RESTORATION

The most common form of localized mesh restoration is *span restoration* (also called *link restoration*) [60], where a centralized or self-organizing re-routing mechanism deploys a collectively coordinated set of replacement paths between the end-nodes of the failed span and effectively "patches" the failure. Spare capacity is seized at the time of restoration only, and will otherwise remain available for use as required to route restoration paths for any failure scenario (or for carrying preemptible unprotected traffic). For each working channel on the failed span, one restoration path must be established between the end-nodes of the failed span, and so long as spare capacity is available to accommodate them, restoration paths can be formed through any number of distinct routes as illustrated in Figure 4.4. Span restoration thus provides a logical detour comprised of a set of replacement path segments around the break that disrupts working paths, without the knowledge of or consideration for the ultimate origin-destination (O-D) nodes of each working path being restored. The end-nodes of the failed span (between which the restoration paths are formed) are called the *custodial nodes* for that particular failure scenario since it is these nodes that act to initiate a restoration response between them.

Span restoration is the mesh technology equivalent to OSPR in that restoration occurs by rerouting between the immediate end nodes of the break. Unlike rings, however, mesh span restoration need not be via a single route (nor via simple two-hop or three-hop routes only); replacement paths can follow many distinct and possibly quite lengthy routes.

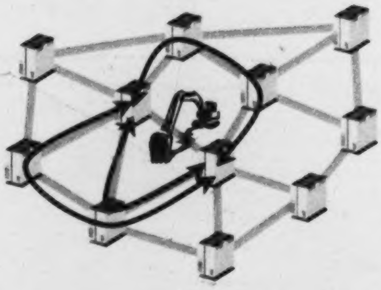


Figure 4.4 – An example of span restoration.

4.4.1.1 DISTRIBUTED VERSUS CENTRALIZED RESTORATION

Span restoration can be achieved by some sort of centralized control mechanism that makes routing decisions and sends commands to the nodes, or through a *distributed restoration algorithm* (DRA) where nodes are allowed to act autonomously and asynchronously. Since a centralized control mechanism has a complete global view of the network's state, including topology and link availability, centralized control is capable of finding an optimal configuration (say, one that minimizes the amount of spare capacity used). However, a centralized approach requires the central database to be continually updated, there exists a single-point of failure, and a diverse telemetry network is required. DRAs, on the other hand, are more robust because they don't need a global view of the network state and aren't vulnerable to a single point of failure, but they won't necessarily find an optimal configuration. Also, many network operators are wary of trusting their networks to such a distributed self-organizing process.

One well-known DRA is the *self-healing network* (SHN) protocol [45], [47], which operates through *statelets* applied to the overhead bytes of each channel in the network. Nodes react autonomously to statelets appearing on the links they are incident on, and by following well-defined rules that all nodes in the network share, neighbouring nodes are able to form cross-connections that ultimately assemble a restoration path between the failed span's end-nodes. In SHN, prior to failure, all nodes transmit null statelets on all links they are incident on. After failure, however, one of the end-nodes (the *sender node*) of the failed span is designated to begin sending statelets indicating, among other things, its own identity and the identity of the other end-node (the *chooser node*) of the failed span. Through a set of simple rules, the statelets propagate through the network from node to node, and each intermediate node (*tandem node*) updates the statelets to indicate that they passed through it. Eventually, the chooser node detects the statelets and initiates reverse linking to trigger cross-connection of the appropriate links, and restoration paths are formed. For details on the SHN protocol, refer to [45] and [47].

4.4.2 SHARED BACKUP PATH PROTECTION

More recently, with the continuing evolution of optical networking and the prominence of IP networks, *shared backup path protection* (SBPP) is emerging as a promising form of network protection [23], [74], [105], particularly in IP networks through the use of MPLS protocols. SBPP is an end-to-end mesh protection mechanism that is at first glance very similar to 1+1 APS in that working traffic between an O-D pair of nodes can be restored over a single pre-defined disjoint backup path. But in SBPP, spare capacity on the backup paths can be shared or used by backup paths from multiple O-D pairs so long as they contain no shared-risk links (i.e., their primary service paths must be diversely routed), greatly reducing capacity redundancy. Demands on working paths that follow physically disjoint routes over the network will not need the restoration capacity simultaneously. This is logically the same scheme as was proposed for ATM Backup VP restoration [69] in the special case where the maximum permissible over-subscription factor, [121], [122], is limited to 1.0.

SBPP and its sharing of backup capacity is illustrated in Figure 4.5. Two node pairs A-B and C-D each route their demands through fully disjoint primary working paths (solid lines), and so there is no single-span failure scenario in which both primary working paths will fail simultaneously. That being the case, both node pairs' backup paths (dashed lines) can share capacity on span X-Y as shown since neither will require use of that spare capacity if the other also needs it (assuming we are dealing with single-span failures only).

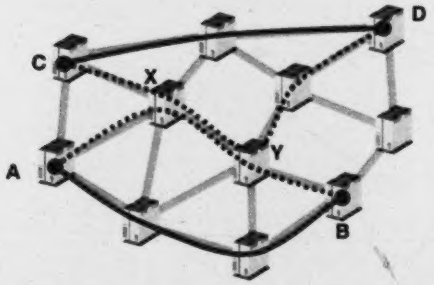


Figure 4.5 – An illustration of shared backup path protection.

SBPP is sometimes called *failure-independent path protection* (FIPP) because the route of the backup path is the same regardless of where a failure arises on the corresponding working path. This simplifies activation and speeds up cross-connection of the backup path, but it foregoes the opportunity to re-use the surviving "stub" portions of the failed path either for the same working demand or for restoration of any other demands that underwent simultaneous failure in the corresponding span cut.

We should note that SBPP is receiving a great deal attention in recent IETF deliberations [74], almost to the point that it has become the de facto standard restoration mechanism in IP networks and some other applications. However, there are drawbacks to SBPP that are for the most part being largely ignored by that community. The biggest disadvantage of SBPP is that every node requires an up-to-date network state database including spare channel sharing relationships throughout the entire network and OSPF topology and capacity information for each span, and it must be simultaneously updated network-wide following every connection set-up or teardown, or any change in network state. Also, when a span failure occurs, potentially very many O-D node pairs may all be brought down simultaneously. While an appropriate network capacity design will ensure that any shared spare capacity will not be needed by more than one O-D pair, each node in the network may be required to perform a large number of simultaneous cross-connections (the SBPP mechanism requires a signalling phase from each tail-end switch to confirm availability of the backup route and to seize and cross-connect capacity at every hop to activate the backup path).

One advantage of SBPP over span-restoration is that if its backup routes are designed to be fully *node*-disjoint from their corresponding working paths, then SBPP is able to protect a network from node failures in addition to single span failures (at least with respect to the traffic transiting the failed node).

4.4.3 PATH RESTORATION

Another form of end-to-end mesh restoration is *path restoration* [62], [63], [121], also called *failure-dependent path protection* (FDPP). As in SBPP, the replacement paths are formed between the origin and destination nodes of all affected lightpaths, but in path restoration, the specific restoration paths used will depend on where in the working path the failure occurred. In true dynamic path restoration, we perform a *multi-commodity maximum flow* (MCMF) type of simultaneous end-to-end re-routing of all O-D node pair demands affected by a failure [62]. Typically, path restoration employs *stub-release* [63], where the surviving stub portions of each affected working lightpath is considered available as spare capacity for re-use by that particular demand's restoration path or by other demands' restoration paths, as needed. The automatic propagation of an Alarm Indication Signal (AIS) in a digital wrapper is a simple and fast means to effect stub release.

Path restoration is illustrated in Figure 4.6. Three node pairs A-B, C-D, and E-F each route their demands through working paths as shown in Figure 4.6(a). The next two panels show two different span failure scenarios and the restoration routes used by each affected demand. In Figure 4.6(b), node pairs A-B and C-D are both affected by failure of span 1 and so the restoration routes by each do not share the same spare capacity. But in Figure 4.6(c), when node pairs A-B and E-F are affected by failure of span 2, A-B can now use a different restoration route than the one it used for restoration of failure of span 1, and that route can now share spare capacity that was previously used by the restoration path for node pair C-D.

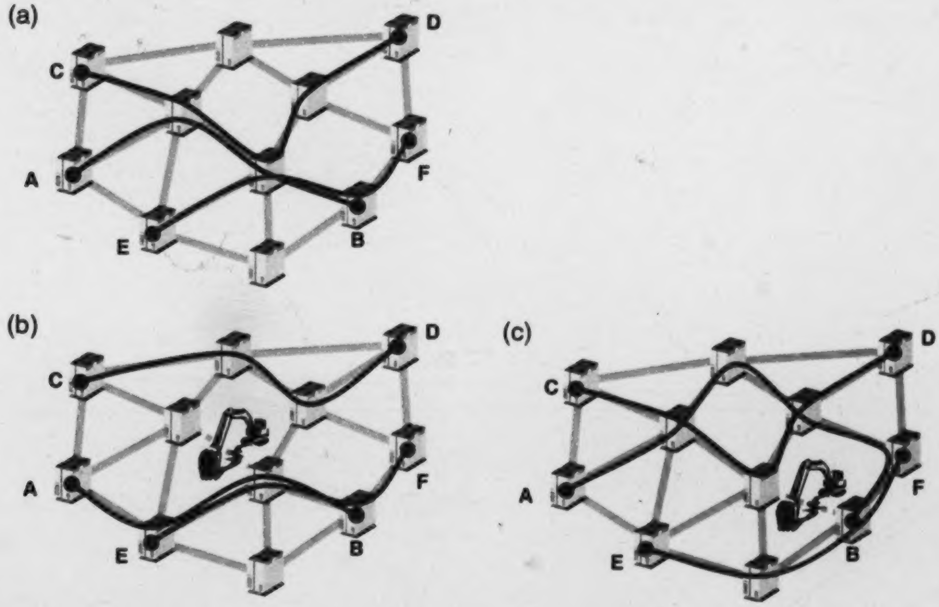


Figure 4.6 – An illustration of path restoration with stub-release.

The main difference relative to SBPP is that there is no single predetermined restoration route for each working path. Rather a collectively optimized re-routing of all failed working paths will occur end-to-end in the presence of the specific failure, the surviving spare capacity following that failure, and the environment of stub release capacity. The restoration routes used by a particular demand for one failure scenario may not necessarily be the same as those used by the same demand for a different failure scenario. Because of this added degree of freedom over SBPP, path restoration is guaranteed to be at least as efficient as SBPP, and when stub-release is used, it is even more capacity efficient. Path restoration can be achieved by a centrally controlled (and possibly pre-planned) re-routing mechanism, or a self-organizing re-routing mechanism similar to one described in [62]. Although path restoration is operationally much more complex than span restoration, it is also fundamentally and provably more capacity-efficient than span restoration as well. And like SBPP, path restoration is also able to protect a network from node failures.

4.4.4 *P*-CYCLES

p-Cycles are a more recent form of network survivability mechanism, combining the speed of ring networks with the capacity efficiency of mesh networks [56], [103], [107], [108]. *p*-Cycles are ring-like pre-configured structures of spare capacity used to protect against failure of *on-cycle spans* (those spans that are a part of the *p*-cycle) and *straddling spans* (those spans whose end-nodes are both on the *p*-cycle, but which are not actually a part of the *p*-cycle itself). Upon failure of a protected span, the *p*-cycle is "broken into" by the restoration mechanism to re-route lightpaths around the failure. The fundamental difference between *p*-cycles and rings, and the source of *p*-cycles' increased efficiency, is in the protection of straddling span failures. A unit-sized ring can only protect against the failure of a single wavelength on each span on the ring itself, but a unit-sized *p*-cycle can protect the failure of a wavelength on each on-cycle span as well as *two* wavelengths on each straddling span, for the same amount of spare capacity. The reason a *p*-cycle can protect two units of working capacity on each straddling span is because if the straddling span fails, one unit can be restored in the clockwise direction around the cycle, and the other can be restored in the counter-clockwise direction.

p-Cycle restoration is illustrated in Figure 4.7. The *p*-cycle is as shown in Figure 4.7(a). For failure of an on-cycle span in Figure 4.7(b), the working capacity on the failed span is re-routed around the *p*-cycle. For failure of a straddling span in Figure 4.7(c), the working capacity on the failed span can be re-routed in either direction around the *p*-cycle. If there are two units of working capacity on the failed straddling span, then one unit can be restored in each direction.

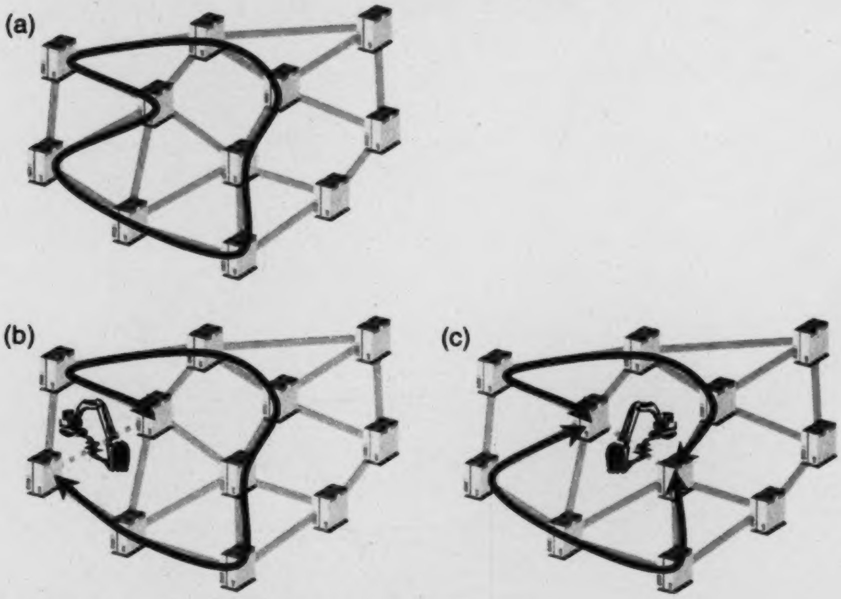


Figure 4.7 – An illustration of *p*-cycle restoration.

While not quite as capacity efficient as other mesh restoration and protection schemes (span restoration, path restoration, SBPP), *p*-cycles provide the same very rapid restoration as rings because they are pre-connected prior to failure. They are significantly more efficient than rings, however, because of their ability to protect straddling spans. A simple comparison of a ring and a *p*-cycle will easily demonstrate how important the protection of straddling spans is to *p*-cycle efficiency. In Figure 4.8(a), a unit-sized ring covering the same eight spans as the *p*-cycle in Figure 4.7 can only protect one unit of working capacity on each of those spans, for a total of eight protected working capacity units. In Figure 4.8(b), however, a unit-sized *p*-cycle covering those same spans not only protects one unit of working capacity on each of the eight on-cycle spans, but also protects two units of working capacity on each of five spans that straddle the *p*-cycle. So for an investment of eight units of spare capacity (same as the ring), the *p*-cycle can protect a total of 18 units of working capacity (2.25 times the protection).

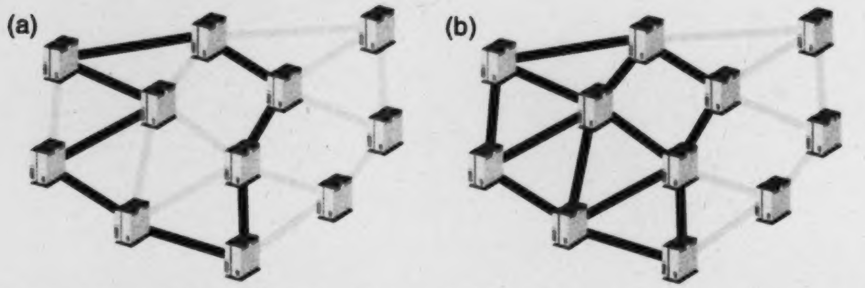


Figure 4.8 – Spans protected by (a) an eight-span ring and (b) a comparable eight-span *p*-cycle.

p-Cycles have another key benefit over rings, in that working paths can be shortest path routed (or otherwise routed as desired) through the network graph; they are not constrained by the *p*-cycle systems used to protect them. In a ring network, on the other hand, working routing must follow the ring structures and inter-ring transitions. In fact, working paths on a straddling span do not even need to be routed within any *p*-cycle at all to be protected as long as each span crossed is a straddling span. And if a *p*-cycle network is designed appropriately, some spans do not require any spare capacity at all, but rather are pure working capacity spans (i.e., 100% revenue-earning). Although *p*-cycles represent a closed-form type of system more similar to rings than to mesh survivability, we generally class it as a form of mesh protection since working signals are allowed to follow shortest path routing.

4.5 PROTECTED WORKING CAPACITY ENVELOPE

Span restoration and *p*-cycles are both forms of ***bearer capacity protection***, where it is the working capacity on a failed span that is protected and restored rather than an explicit protection and restoration of the affected working paths themselves. A new networking concept called the ***protected working capacity envelope*** (PWCE) [49] makes bearer capacity protection mechanisms particularly attractive relative to SBPP and path restoration, which restore failed working lightpaths end-to-end. In the PWCE framework, if we properly design a network so that pre-determined amounts of working capacity are fully restorable for single span failures (say, using span restoration), then any working lightpaths that are routed over that working capacity are by extension also fully restorable. Those pre-determined amounts of working capacity can be considered to be an envelope of protected working capacity within which lightpaths can be provisioned and their survivability will be assured. Re-routing is completely transparent to the lightpaths themselves and there are no explicit provisioning operations required to provide for their survivability. The service layer merely routes over a shortest path, designates the protection priority or class, and receives a confirmation that the path is protected when routed within the current envelope of protected working span capacities.

The PWCE concept can be particularly useful in a dynamic provisioning environment, where lightpath demands need to be routed and provisioned (and released) in real time. Using SBPP, which has become the de facto standard survivability mechanism for dynamic provisioning of protected lightpaths, not only does the working lightpath need to be provisioned in real time, but that process is also tasked with explicitly arranging the shared capacity protection path at service set-up as well. Under current SBPP proposals, this requires a global OSPF-TE type of topology and resource database in every node including a database of current network-wide backup link sharing arrangements, making this scheme intensively dependent on Internet-type dissemination of global state and database synchronization issues. Under the PWCE paradigm (using either span restoration or *p*-cycles), however, lightpath provisioning is greatly simplified to a much more scalable situation where protection is inherent, so long as the working capacity is present to route the new path. There is no requirement to explicitly arrange a shared disjoint backup path for every individual provisioning operation, or to update the network-wide state for each new protection path set-up or takedown. The capacity used for dynamic provisioning is *itself* protected, so any service provisioned over available working capacity is inherently also protected with no further requirements. In contrast to SBPP, where explicit arrangements for protection are referred into the service provisioning problem for every path, span or *p*-cycle restoration in a PWCE environment offers an alternative operational paradigm that may appeal to many operators: that of provisioning over protected capacity versus explicitly provisioning protection.

4.6 RESTORATION VERSUS PROTECTION

Throughout the preceding discussions on survivability, we have used the terms *restorability* and *protection* to refer to various survivability mechanisms and actions, apparently interchangeably. However, in general, the term *protection* is used for schemes where the switching actions required post-failure are pre-defined and spare capacity is often dedicated to cover a specific set of failure scenarios such as in 1+1 diverse-routed protection, or path- or line-switched rings. In pure protection mechanisms, even cross-connection is unnecessary when the signal is switched to the backup path; the backup paths are pre-configured into a pre-tested and ready-to-use state. However, the term protection is also used to refer to schemes such as SBPP where the protection route is known ahead of time, but multiple hops of capacity seizure and cross-connection remain to be accomplished in real time.

Restoration, on the other hand, generally refers to mechanisms where replacement paths do not need to be pre-defined and where a network-wide allocation of spare capacity is not dedicated to any specific failure(s) but is configured as needed to restore affected carrier signals as failures arise. In their purest form, restoration mechanisms determine replacement paths, seize spare capacity, and form the appropriate cross-connections all in real time as a response to a failure, either through a centralized mechanism or a distributed protocol. However, depending on the specific implementation, some restoration mechanisms can carry out pre-planning exercises, and even some amount of pre-configuration is possible. Restoration by its very nature is adaptive to unexpected changes in the network state, and as such, will typically exhibit better availability[7] than protection mechanisms.

One common impression that many in the industry have is that protection mechanisms are fast and restoration mechanisms are slow, but this is not necessarily correct, and so we should take care not to over-simplify or over-emphasize the importance of such classifications. In fact, any restoration mechanism can be modified into a corresponding pre-planned protection mechanism by way of constantly updated

---

[7] *Availability* refers to the proportion of time a device (say a specific service path or perhaps even the network as a whole) is in a functional operating state [5]. This should not be confused with *reliability*, which is the probability that a device will perform as required for a specified period of time.

distributed pre-planning (DPP) actions. Under DPP, either a distributed or centralized restoration mechanism is used to repeatedly and autonomously pre-plan fast protection reactions to any single failure. These so-called pre-plans record the simplest possible local information of only what spare links each node needs to cross-connect, as fast as possible, in response to a simple flooding advertisement of the failure notification. Even if finding replacement paths is slow, it is of little concern because distributed pre-planning does this *before* failure and can create (and frequently update) the protection plans. Thus the speed of finding replacement paths is completely decoupled from the real time speed of reaction. The concept is described more fully in [46] and [47]. The significance of DPP is that it allows, say, span restoration to provide for a fast pre-planned protection reaction to any single failure. However, it also has the added advantage that the option to fall back to on-demand use of the adaptive real-time span restoration process is always there if needed to maximize the restorability in the face of dual-failures, or in any situation where the pre-planned response did not yield the desired recovery level. This permits a very robust integrated strategy of *first-failure: (pre-planned) protection, second-failure: (adaptive) restoration*. In contrast, with a pure protection scheme, if a prior failure on another span thwarts the access to the one pre-planned backup required by a second failure, there is no automatic secondary response that can recover adaptively in any reasonable amount of time.

4.7 REDUNDANCY AS A MEASURE OF NETWORK EFFICIENCY

In some of the discussions of the previous sections, we have used the term *redundancy*. Although we described it briefly in Section 4.2 with reference to 1+1 APS as being equivalent to the ratio of the total wavelength-kilometres required for backup routing relative to the working wavelengths transported over their shortest paths, that description is not entirely accurate in the general sense. To be more precise, we define the *capacity redundancy* (R_{cap}) of a network as the total amount of spare capacity over all of its spans divided by the total amount of working capacity over all of its spans, as shown in equation (4.1). Redundancy is a commonly used measure of a network's efficiency, and clearly, the lower the redundancy, the better, since a low redundancy implies a relatively small amount of spare capacity is needed to protect a network's working flows.

$$R_{cap} = \sum_{\forall j \in S} s_j \Big/ \sum_{\forall i \in S} w_i \tag{4.1}$$

Alternatively, when performing network design optimization based on capacity costs, rather than unit capacities, we can calculate *capacity cost redundancy* (R_{cost}) of a network as the ratio of the total cost of spare capacity in the network to the total cost of working capacity, as shown in equation (4.2). When all of a network's span costs are equivalent (i.e., $c_1 = c_2 = c_3 = c_4 = \ldots$), then $R_{cost} = R_{cap}$, but in general, this will not be so (and there is no simple means of expressing R_{cost} in terms of R_{cap}, or vice versa).

$$R_{cost} = \sum_{\forall j \in S} \left(s_j \cdot c_j\right) \Big/ \sum_{\forall i \in S} \left(w_i \cdot c_i\right) \tag{4.2}$$

Many people are already familiar with the well-known $1/\left(\bar{d}-1\right)$ lower bound on redundancy in a span-restorable network, [25], [28], [46], and have an intuitive understanding of it as a basis for how redundancy should drop as connectivity increases. This topological lower bound is derived from arguments about the conditions for restorability local to any one isolated node within a span-restorable network. We know experimentally that, at least for span-restorable networks, the restoration flow is usually limited by the total spare capacity incident to one of the end-nodes of the failed span. More specifically, in a span-restorable network there must be at least as much spare capacity on all surviving spans incident on a failed span's end-node as there is working capacity on the failed span itself. Such end-node limiting suggests that the conditions at the end-nodes may therefore provide the basis for a good bound, at least for the span-restorable case. This observation can only provide a lower bound, however, because it predicts *necessary* but not necessarily *sufficient* capacity conditions for the network as a whole to be fully restorable. Somewhat similar end-node limited conditions can also be considered for path-restorable networks where the end nodes are the O-D nodes of each demand pair affected by a failure [63].

We make use of Figure 4.9 in deriving the $1/\left(\bar{d}-1\right)$ lower bound. In the leftmost panel, a degree-*d* node is incident on *d* spans with working capacities indicated. The spans are numbered in order of largest to smallest working capacity. Span 1 is therefore the span with the most working capacity, w_1, span 2 has the next most working capacity, w_2, and so on. In other words, $w_i \geq w_{i+1}$. Span 1 also has s_1 units of spare capacity on it, span 2 has s_2 units of spare capacity, etc., as indicated in the right-most panel of Figure 4.9.

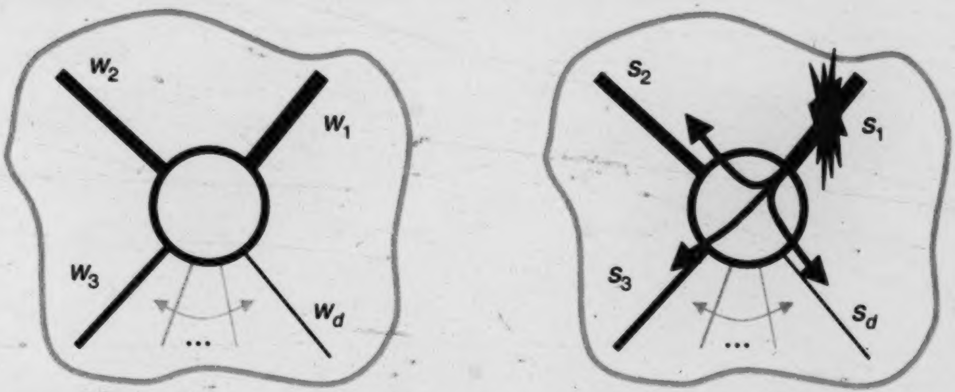


Figure 4.9 – The basis for a derivation of a topological lower bound on redundancy in a span-restorable network.

If we consider the failure of span 1, then clearly there must be enough total spare capacity available on all of the other spans to permit restoration of w_1 units of working capacity on span 1. In other words, we know that $\sum_{\forall j \in \{1 \ldots d\} | j \neq 1} s_j \geq w_1$. Similarly, in the absence of higher network-level considerations that may still add more spare capacity, each other span requires for its restoration that the total amount of spare capacity on surviving spans meets or exceeds its own working capacity. So for failure of span 2, we have $\sum_{\forall j \in \{1 \ldots d\} | j \neq 2} s_j \geq w_2$, etc., and in general, for failure of any span i, $\sum_{\forall j \in \{1 \ldots d\} | j \neq i} s_j \geq w_i$. It follows that in the best case from an efficiency standpoint, every span $i \in \{1 \ldots d\}$ would have working capacity $w_i = w_1$, and spare capacity would be evenly distributed on all spans $j \in \{1 \ldots d\}$ so that $s_1 = s_2 = \ldots = s_j$, and would be no more than exactly needed for restoration of each span failure. If that is the case, and since $\sum_{\forall j \in \{1 \ldots d\} | j \neq 1} s_j \geq w_1$, then we will have $w_1 = s_j \cdot (d-1)$, or by extension, $s_j = \frac{w_1}{d-1}$.

So we calculate the capacity redundancy at that node to be:

$$R_{cap} = \sum_{\forall j \in \{1 \ldots d\}} s_j \Big/ \sum_{\forall i \in \{1 \ldots d\}} w_i = \left(d \cdot \frac{w_1}{d-1} \right) \Big/ (d \cdot w_1) = 1/(d-1) \tag{4.3}$$

Other methods of estimating a span-restorable network's redundancy are given in [25], and a network-wide lower bound on redundancy is developed in [104].

4.8 OTHER ISSUES AND CHALLENGES IN NETWORK SURVIVABILITY

While the scope of this thesis is generally limited to dealing with failures representative of entire span cuts or, in some cases, entire node failures in a transport network, there are other issues related to network survivability that are not addressed herein. For instance, deliberate and malicious damage or outages caused by hackers, denial of service attacks, computer viruses, etc., as well as other issues such as software reliability, router errors, and equipment faults are definite concerns to network operators and will have significant impacts on their networks' availability or reliability, [65], [76]. A quick scan of recent CA*net 4 outage reports [12] will reveal that failures, service interruptions, service degradations, and the like are quite frequent but most are a result of causes such as those just mentioned, rather than entire span cuts. Large-scale power failures and natural phenomena (i.e., flooding, ice storms, earthquakes, etc.) are also factors in network survivability, and although they are quite rare, those situations could cause widespread and catastrophic network failures requiring vastly different approaches to deal with than those discussed herein.

Another factor to consider is the duplication of restoration functionality available at the various layers of the service layer stack (see Section 3.1). In the IP layer, for example, survivability can be achieved by using a routing protocol such as OSPF to update the IP routing tables so that traffic is routed around the failure [96]. And ATM restoration can be via backup virtual path protection, which is very similar to SBPP discussed in Section 4.4.2 [69]. One aspect of restoration in some of these higher layers is that the concept of *over-subscription*, has meaning. Here, the ratio of available spare capacity to restoration route allocations can be greater than one, meaning there is the potential for overload [48]. Furthermore, while the survivability methods dealt with in this thesis are described using transport networking language, the concepts are valid in other contexts as well. *p*-Cycles, for instance, were developed and initially described with optical transport networking in mind, but the basic mechanism has been shown to be applicable in IP layer restoration as well [108]. It is even possible to coordinate network protection between the various layers [96]. One difficulty with doing this is that there is the potential for each layer to detect and react to a failure independently, resulting in unnecessary action in some layers. Contention issues could also arise, delaying (or preventing) the overall restoration of the failure and/or making inefficient use of resources [96]. If coordinated properly, say, by scheduling restoration responses differently in each layer, multiple-layer restoration could allow the various mechanisms to compliment each other. Transport layer restoration might then be used to deal with one failure, while a second failure (overlapping in time with the first) might be dealt with in the IP layer where over-subscription is permitted.

More comprehensive discussions of other network survivability, reliability, and availability issues are available in [6], [48], [65], [76], [96], and [111]. For this thesis, however, the scope of our work will be limited to dealing with span cuts and node failures (or their equivalents) within a transport network.

CHAPTER 5

MESH-RESTORABLE TRANSPORT NETWORK DESIGN AND OPTIMIZATION

The work in this thesis generally deals with the problem of designing mesh-restorable transport networks so that the costs of capacity required in the network have been minimized, but subject to constraints ensuring full working lightpath routing and 100% restorability to specifiable failures (typically single spans only). Given a transport network graph topology, a set of working lightpath demands, and possibly other criteria that must be met, the network design problems addressed herein seek to determine the minimum cost capacity distribution (i.e., the number of working and/or spare capacity links on each span) that allow full working lightpath routing while simultaneously guaranteeing full restorability under whichever survivability scheme chosen. This type of network design problem can take the form of either a single-stage or a two-stage problem. In the single-stage problem, working lightpaths are routed first by some method (and usually follow shortest path routing), and then a *Spare Capacity Allocation* (SCA) problem optimally determines restoration routing and dimensions the *reserve network* – the spare capacity distribution overlaid on the same network topology that carries work routing. In the two-stage network design problem, working and restoration routing (and subsequent working and spare capacity determination) are performed *jointly*, so that the total network capacity cost is optimized. This joint optimization method is generally called *Joint Capacity Allocation* (JCA). The aspect of jointness allows working paths to be routed in other than a shortest path manner so that, in conjunction with the spare capacity needed for restoration, the total (working plus spare) capacity requirement is minimized.

Other possible design problems could seek to determine the working and restoration routing so as to reduce blocking probabilities (i.e., the probability that a newly arriving demand relation is routable and/or restorable), maximize network restorability given a pre-determined capacity arrangement, maximize network availability given a pre-determined capacity budget, etc. While these are all valid and important design objectives, they are beyond the scope of this thesis, and only the SCA and JCA design problems posed above (and some variations on them) are considered and discussed herein.

There are various methods for implementing SCA and JCA design problems, and the exact model used will depend in part on the restoration mechanism employed within the network. For instance, the SCA design problem for a span-restorable network is generally a form of non-simultaneous single-commodity capacity allocation problem. Early work on similar problems was to support time-varying network flow patterns [106]. The main difference in applying that work to one for span restoration is that we effectively delete one edge of the network graph for each of the non-simultaneous flow requirements, thereby simulating edge failures.

Other work that was done specifically for transport network restoration began with a proposed linear programming representation of the SCA problem based on a *min-cut max-flow* model [100]. Here, spare capacity is assigned so that for each possible span failure, the *minimum spare capacity cutset* on the surviving edges is sufficient for full restoration of the failed span's working capacity. In this context, we define a cutset as any set of edges whose removal from the network would result in the end-nodes of the failed span being in two disconnected components of the graph, and the minimum spare capacity cutset is the one whose edges carry the minimum total spare capacity. It is shown in [100] that the maximum flow possible between any two nodes in a network is equivalent to the minimum spare capacity cutset, and so by ensuring that the minimum spare capacity cutset for each span failure scenario is at least as large as the working capacity on the failed span, we can guarantee full restorability. One technical challenge with this approach is that the number of possible cutsets in a network is $O(2^S)$, and so enumerating all cutsets becomes computationally infeasible. However, finding a suitably small subset of cutsets that fully constrain the solution while also permitting an optimal (or near-optimal) capacity design is difficult. In [100], the approach is to use a constraint generation technique where successive solutions of an LP detect and add missing constraints, which are discovered by testing the resultant design for restorability at each stage. Enhancements in [114] use an efficient algorithm to discover relevant new cutsets and a "path table" data structure to allow for fast restorability testing. For more information on the min-cut max-flow solution methods, refer to [48], [100], and [114].

Herzberg et al [59], [60] proposed an *arc-path* LP formulation for the SCA problem in a span-restorable network. Here the network graph topology is first pre-processed to find all distinct logical routes that are eligible for use in the restoration of each failure scenario. Spare capacity values on each span are sized to support the largest assignment of simultaneous restoration flows over the eligible restoration routes crossing each edge in the network graph over all non-simultaneous failure scenarios such that the total spare capacity is minimized. The number of distinct routes possible is $O(2^S)$, but the complexity of the problem can be greatly reduced in practice by reducing the number of eligible routes provided to the problem (through the use of route hop-limits to restrict eligible route lengths) with little or no loss of solution quality, [59], [60]. This approach also gives a detailed explicit specification of restoration routes and flows, while the min-cut max-flow approach does not. Another practical advantage of the arc-path method is that restoration route properties can be under direct engineering control to limit such properties as length, hop count, signal loss, etc., for each failure scenario, while the min-cut max-flow approach does not.

The SCA design problem can also be expressed in the form of a set of *transhipment* or *network flow* LP problems [119]. In a network flow problem, *supply nodes* and *demand nodes* act as sources and sinks of a *commodity* (i.e., a lightpath demand), and *transhipment nodes* simply act as intermediaries that pass along any of the commodity it receives to other nodes. Like the arc-path model, the network flow model allows us to explicitly specify the amount of flow over restoration routes (although it is somewhat more complicated to do so), but it does not allow for easy control of restoration route properties.

5.1 ARC-PATH ILP FORMULATIONS

Unless otherwise stated, the network design methods used in this thesis follow the arc-path type of formulation. The exact structure of the models differs depending on the survivability mechanism of the network we are designing, but in general, all such models require an explicit enumeration of a set of eligible restoration routes (and in the case of joint designs, working routes as well). Descriptions of the arc-path models used for each of the various survivability mechanisms discussed herein follows.

5.1.1 SPAN RESTORATION

The basic arc-path formulation of the SCA problem in a span-restorable network uses the following notation:

Sets:

- S is the set of spans in the network, and is typically indexed by i when referring to a failure span and j when referring to a surviving span.
- P_i is the set of all distinct eligible routes available to carry restoration flow for failure of span i, and is typically indexed by p.

Input Parameters:

- c_j is the cost of each unit of capacity (working or spare) on span j.
- w_i is the amount of working capacity to be protected on span i, and arises from a prior routing of working routes (typically shortest path routing).
- $\delta_{i,j}^{p} \in \{0,1\}$ is a parameter that encodes restoration routes. If $\delta_{i,j}^{p} = 1$, restoration route p used for restoration of span i crosses span j. If $\delta_{i,j}^{p} = 0$, restoration route p used for restoration of span i does not cross span j.

Decision Variables:

- $s_j \geq 0$ is the amount of spare capacity that is placed on span j.
- $f_{i,p} \geq 0$ is the amount of restoration flow assigned to restoration route p for failure of span i.

The ILP formulation of the span-restorable SCA problem itself is then expressed as follows.

Minimize $$\sum_{\forall j \in S} c_j \cdot s_j \qquad (5.1)$$

Subject to: $$\sum_{\forall p \in P_i} f_{i,p} = w_i \qquad \forall i \in S \qquad (5.2)$$

$$\sum_{\forall p \in P_i} \delta_{i,j}^{p} \cdot f_{i,p} \le s_j \qquad \forall i,j \in S \,|\, i \neq j \tag{5.3}$$

The objective function in equation (5.1) seeks to minimize the total cost of placing spare capacity in the network. For simplicity, we usually equate c_j with the length of the span (the Euclidean distance between the end-nodes of the span as drawn in the network graph), and in general, this cost can be thought of as representing the actual costs of fibre, rights-of-way, amplifiers, etc., most of which are at least partially distance-dependant. The constraints in equation (5.2) ensure that the total restoration flow assigned to all eligible restoration routes for failure of span i is sufficient to fully restore all of the working capacity on the failed span. Equation (5.3) places enough spare capacity on each surviving span j to accommodate the total restoration flow assigned to all restoration flows crossing it for restoration of any failed span i. More specifically, each s_j quantity in equation (5.3) is determined by the largest sum of simultaneously imposed restoration flows over span j, over the set of all non-simultaneous failure scenarios not involving span j itself as a failed element. Thus, the spare capacity assigned to each span j could arise from any of a number of different finite-flow sub-problems, there being one such sub-problem for each span failure scenario. Each individual failure scenario, taken in isolation, is similar to a two-terminal minimum cost network flow problem [119], but the formulation above couples them all together under the objective of minimum sparing. It is for this reason that the constraints in equation (5.3) are not strict equalities; the spare capacity required on span j for one particular failure scenario may exceed that required for another failure scenario. The overall result of the formulation is a minimum sum of span-wise maximum quantities of the restoration flows assigned to each span.

To correspond to a WDM optical network where a unit of capacity represents an individual wavelength, the SCA problem would be solved as a pure ILP, with all s_i and $f_{i,p}$ variables taking on strictly integer values only. In [59] and [60], the SCA problem was solved as an LP for runtime considerations (ILPs are much more difficult to solve than LPs), and rounding and variable adjustments were used to approximate the optimal integer solution. This can usually be done with only minor loss of optimality, but in practice, however, with today's much faster computers and more efficient ILP solvers, the problem can often be solved directly as an ILP even for reasonably large sizes. And as discussed in [3] and [117], as long as the capacity variables (s_i) are integral, the integrality requirement on the underlying flow variables ($f_{i,p}$) can be relaxed without affecting optimality or feasibility. To reduce solution runtimes, we therefore solve the SCA problem (and unless otherwise stated, all other network design problems discussed in this thesis) as an MIP with integer capacity variables and real flow variables.

The SCA formulation for a span-restorable network can easily be modified to the JCA formulation in order to perform joint working and spare capacity optimization. To do so, the prior w_i input parameters become output variables, we modify the objective function, and we add new set, parameter, and variable notation as well as two new constraints to ensure the routing of working demands and adequate working capacity to support them. The added notation for the joint problem is:

New Sets:

- D is the set of working lightpath demands, and is typically indexed by r.
- Q^r is the set of all distinct eligible routes available to carry working path routing for demand relation r, and is typically indexed by q.

New Input Parameters:

- d^r is the number of lightpath demand units for demand relation r.
- $\zeta_j^{r,q} \in \{0,1\}$ is a parameter that encodes working routes. If $\zeta_j^{r,q} = 1$, working route q used for demand relation r crosses span j. If $\zeta_j^{r,q} = 0$, working route q used for demand relation r does not cross span j.

New Decision Variables:

- $g^{r,q} \geq 0$ is the amount of working flow assigned to working route q used for demand relation r.
- $w_i \geq 0$ is the amount of working capacity that is placed on span i (this was formerly an input parameter).

In addition to the new notation, all previous notation from the SCA formulation of the problem remains. In order to consider working capacities in addition to spare capacity, we also make the following change to the objective function:

$$\text{Minimize} \quad \sum_{\forall j \in S} c_j \cdot (s_j + w_j) \tag{5.4}$$

Finally, we add two new constraint sets as follows:

$$\sum_{\forall q \in Q^r} g^{r,q} = d^r \qquad \forall r \in D \tag{5.5}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r} \zeta_j^{r,q} \cdot g^{r,q} = w_j \qquad \forall j \in S \tag{5.6}$$

Equation (5.5) is the working routing equivalent to the restoration-related constraint set in equation (5.2). It ensures that the total working flow assigned to all eligible working routes for demand relation *r* is sufficient to fully route it. Note that as it is written, there is nothing in equation (5.5) (or other constraint sets) preventing working flow from being split amongst multiple working routes. Equation (5.6) sizes the working capacities on each span in much the same way that equation (5.3) does so for spare capacity. The main structural difference between those two constraint sets is that in equation (5.6), the working flow for each demand relation is applied to each span *at the same time*, hence the double summation. When sizing spare capacity in equation (5.3), on the other hand, restoration flow is applied separately for each span failure scenario. Also, we use an equality here, rather than an inequality as in equation (5.3). This is because working capacity placement is strictly equivalent to that required to carry all lightpath demands, which are all routed simultaneously, while spare capacity in equation (5.3) is sized to accommodate individual failure scenarios separately, some of which may require more or less spare capacity than others on any particular span. Besides the change to the objective function and the addition of the two new constraint sets, the constraints in equations (5.2) and (5.3) remain a part of the JCA formulation. While the span-restorable SCA problem can also be solved using the JCA formulation, by simply providing only a single eligible working route (say, the shortest) for each demand relation, it is typically solved using the separate formulation provided earlier.

Many variations on the formulations above are possible. The *capacity only* formulation is one such version where the goal is to minimize the number of working and/or spare capacity units required, rather than the cost of such capacity. It can be modelled in two ways, either by simply setting all c_j capacity cost values to the same value ($c_j = 1 \quad \forall j \in S$ is typical), or actually removing the c_j capacity cost parameter from the objective function. In the latter case, equations (5.1) and (5.4) are replaced by equations (5.7) and (5.8), respectively:

$$\text{Minimize} \quad \sum_{\forall j \in S} s_j \tag{5.7}$$

$$\text{Minimize} \quad \sum_{\forall j \in S} s_j + w_j \tag{5.8}$$

Another common variation on the formulation is one where the network does not necessarily have to provide full (i.e., 100%) restoration for each failure scenario. By introducing a new input parameter, $0 \le l_i \le 1$, which represents the proportion of working capacity on span *i* that must be restorable in the event of that span's failure (i.e., it's *level of restoration*), we can replace equation (5.2) with equation (5.9), below. The right hand side of this new equation now effectively requires sufficient restoration flow over all eligible restoration routes to restore a specified proportion of the failed span's working capacity. If integrality is strictly asserted on the $f_{i,p}$ flow variables, then it may also be necessary to change the equality in equation (5.9) to an inequality (≥) since fractional amounts of required working capacity protection may result.

$$\sum_{\forall p \in P_i} f_{i,p} = w_i \cdot l_i \qquad \forall i \in S \tag{5.9}$$

In equation (5.3), working capacity requirements are calculated as the summation of the products of working flow and a {0,1} binary parameter encoding whether a particular route is crossing a particular span, and similarly for equation (5.6) in calculating spare capacity requirements. However, we can also encode whether a route crosses a span by declaring a set $S_q \subseteq S$, which is the set of spans in route *q*, and which is just as easily generated in pre-processing as the $\delta_{i,j}^{p}$ and $\zeta_j^{r,q}$ parameters. If that is the case, then $\sum_{\forall p\in P_i} \delta_{i,j}^{p} \cdot f_{i,p}$ is equivalent to $\sum_{\forall p\in P_i | j\in S_p} f_{i,p}$ and $\sum_{\forall r\in D}\sum_{\forall q\in Q^r} \zeta_j^{r,q}\cdot g^{r,q}$ is equivalent to $\sum_{\forall r\in D}\sum_{\forall q\in Q^r | j\in S_q} g^{r,q}$. We can then replace equations (5.3) and (5.6) with equations (5.10) and (5.11), respectively. While the two pairs of equations are equivalent, some readers find that this approach is more intuitive, and depending on the model (for instance the SBPP and path restoration models in Sections 5.1.2 and 5.1.3), it may also simplify its expression.

$$\sum_{\forall p\in P_i | j\in S_p} f_{i,p} \le s_j \qquad \forall i,j\in S | i\neq j \qquad (5.10)$$

$$\sum_{\forall r\in D}\sum_{\forall q\in Q^r | j\in S_q} g^{r,q} = w_j \qquad \forall j\in S \qquad (5.11)$$

5.1.2 SHARED BACKUP PATH PROTECTION

In the above formulations for span-restorable network design, only one set of eligible restoration routes need be considered for any given failure scenario since only a single span fails at any given time, and (due to the restoration mechanism) only one commodity requires rerouting. The formulation for SBPP, on the other hand, needs to provide for rerouting of multiple commodities simultaneously, depending on the specific failure scenario and the lightpaths affected by it. This requires significant structural differences in the SBPP formulation compared to the span restoration formulation.

In addition to the notation from Section 5.1.1, we add the following new notation:

New Sets:

- R_r is the set of all distinct eligible routes that can be used either for primary or backup routing of demand relation *r*. It is typically indexed by *p* if we consider it for primary routing, or *b* if we consider it for backup routing.
- $R_r^j \subseteq R_r$ is the set of all distinct eligible routes that can be used either for primary or backup routing of demand relation *r*, and which cross span *j*.
- $S_q \subseteq S$ is the set of spans in route $q\in R_r$, as described above.

New Decision Variables:

- $x_r^b \in \{0,1\}$ is a variable that encodes the assignment of backup routes. If $x_r^b = 1$, then backup route *b* is assigned for use to protect demand relation *r*. If $x_r^b = 0$, then backup route *b* is not assigned for use to protect demand relation *r*.
- $y_r^p \in \{0,1\}$ is a variable that encodes the assignment of primary routes. If $y_r^p = 1$, then primary route *p* is assigned for routing of demand relation *r*. If $y_r^p = 0$, then primary route *p* is not assigned for routing of demand relation *r*.
- $z_r^{p,b} \in \{0,1\}$ is a variable that jointly encodes the assignment of backup routes and primary routes, and acts as a proxy to the product of x_r^b and y_r^p. If $z_r^{p,b} = 1$, then primary route *p* and backup route *b* are both assigned for use by demand relation *r*. If $z_r^{p,b} = 0$, then at least one of primary route *p* and backup route *b* is not assigned for use by demand relation *r*. If $z_r^{p,b}$ did not exist as a separate variable, then equation (5.19) below would need to contain a product of two variables, which would make this formulation non-linear.

The SBPP JCA model is then expressed as follows:

$$\text{Minimize} \quad \sum_{\forall j \in S} c_j \cdot (s_j + w_j) \tag{5.12}$$

$$\text{Subject to:} \quad \sum_{\forall p \in R_r} y_r^p = 1 \qquad \forall r \in D \tag{5.13}$$

$$\sum_{\forall r \in D} \sum_{\forall p \in R_r^j} y_r^p \cdot d^r = w_j \qquad \forall j \in S \tag{5.14}$$

$$\sum_{\forall b \in R_r} x_r^b = 1 \qquad \forall r \in D \tag{5.15}$$

$$x_r^q + y_r^q \le 1 \qquad \forall r \in D \quad \forall q \in R_r \tag{5.16}$$

$$x_r^b + y_r^p \le z_r^{p,b} + 1 \qquad \forall r \in D \quad \forall p, b \in R_r \tag{5.17}$$

$$\sum_{\forall b \in R_r | i \in S_b} x_r^b \ge y_r^p \qquad \forall i \in S \quad \forall r \in D \quad \forall p \in R_r^i \tag{5.18}$$

$$\sum_{\forall r\in D}\sum_{\forall b\in R_r^i}\sum_{\forall p\in R_r^j\left|S_p\cap S_b=\varnothing\right.} z_r^{p,b}\cdot d^r \le s_i \qquad \forall i,j\in S\left|i\ne j\right. \tag{5.19}$$

The objective function (5.12) minimizes the total cost of working and spare capacity needed to provide full primary and backup routing of all demands, and is the same objective function as that for the span-restorable JCA model in equation (5.4). The constraints set in equation (5.13) ensures that there is exactly one primary (working) route for each demand relation *r*. In the case where a network operator may wish to allow demands for a single relation to be routed over more multiple routes as needed for efficiency improvements, then that demand relation can be expressed as individual unit-sized demand relations instead. For instance, if demand relation *D09* represents three lightpaths between nodes *N04* and *N16*, then expressing it as three separate demand relations (say, *D09a*, *D09b*, and *D09c*) of one lightpath each between nodes *N04* and *N16* would allow (but not necessarily force) each to follow distinct primary routes (and backup routes). If we did not express equation (5.13) in this manner, but instead used a variable representing the number of lightpath demand units assigned to each route as we did in the span-restorable models, then later in equation (5.19) when we calculate the amount of spare capacity required on each span, we would not be able to easily avoid a non-linearity in the equation. Equation (5.14) determines the amount of working capacity required on each span to accommodate the primary routing; the working capacity on a span is equivalent to the sum of the number of lightpath demand units of each demand relation *r* whose primary route *p* crosses the span. Equation (5.15) ensures that there is exactly one backup (protection) route assigned for each demand relation *r*. In equation (5.16), any given route *q* can only be assigned as the primary route or the backup route for a demand relation *r* (or neither), but not both. While this constraint is not strictly required because later constraints will ensure span disjointedness between any demand relation's assigned primary and backup routes, this added-value constraint helps to more directly confine the feasible solution space of the ILP. Equation (5.17) assigns a value to the $z_r^{p,b}$ variables, which are used in the constraints set that follows. If $x_r^b=0$ and $y_r^p=0$ (neither routes *p* nor *b* are assigned as the primary and backup route, respectively, for demand relation *r*), then $z_r^{p,b}$ will be allowed to equal 0, and since $z_r^{p,b} = 0$ can only ever decrease spare capacity costs (via the combination of other constraints in the model), then that is the value it will take. On the other hand, if $x_r^b = 1$ and/or $y_r^p = 1$ (at least one of routes *p* or *b* are assigned as the primary and backup route, respectively, for demand relation *r*), then $z_r^{p,b}$ will have to equal 1 for the constraint to be satisfied. Effectively, this is equivalent to $z_r^{p,b} = x_r^b \cdot y_r^p$, but since that is a non-linear equation, then we must express it as we have; its purpose will become apparent in the discussion of equation (5.19), below.

The constraints in equation (5.18) ensure that for each eligible route *p* for demand relation *r* that crosses failed span *i*, if that route is assigned as the primary route (i.e., $y_r^p = 1$), then that demand relation must be assigned a backup route *b* that does not cross the failed span (i.e., $i \notin S_b$). Finally, in equation (5.19), sufficient spare capacity is assigned to each surviving span *j* to accommodate all backup route assignments that simultaneously cross that span for each failure scenario. More specifically, the spare capacity on surviving span *j* must equal or exceed the sum of the number of lightpath units of all demand relations *r* that are assigned backup route *b*, which crosses span *j* and primary route *p*, which crosses failed span *i*, where routes *p* and *b* are span disjoint (i.e., $S_p \cap S_b = \varnothing$). If route *p* is not assigned as the primary route for demand relation *r* and/or route *b* is not assigned as the backup route, then from equation (5.17), $z_r^{p,b} = 0$, and so the only combination of eligible primary and backup routes that will contribute to the spare capacity of span *j* is the pair that is actually assigned.

As in the span-restorable design problems, we enforce integrality on the w_j working capacity and s_j spare capacity variables to correspond to WDM optical networking where a unit of capacity represents an individual wavelength. We must also enforce integrality on the x_r^b, y_r^p, and $z_r^{p,b}$ variables since they must strictly be either equivalent to 0 or 1 in order to have any meaning. Since there are no other output variables in this model, we must therefore solve the SBPP JCA design problem as a pure ILP.

In much the same way that the span-restorable SCA design problem could be solved by using the span-restorable JCA model, we can solve the SBPP SCA design problem with only a simple modification to the SBPP JCA model. To do so, values for the y_r^p variables encoding whether an eligible route p is a primary route for demand relation r or not are assigned as inputs so that $y_r^p = 1$ for, say, the shortest eligible route p assigned to be the primary working route for demand relation r, and $y_r^p = 0$ for all other eligible routes for demand relation r. Equation (5.13) becomes redundant but still holds, and all other constraint sets remain unaffected. In these circumstances, it would also be wise to ensure that eligible route sets provided for each demand relation contain only those routes that are span disjoint from the primary route, so as not to unnecessarily add to the problem's complexity by including routes (and all the accompanying parameters, variables, and constraints) that could not possibly be chosen as backup routes.

An alternative model of the SBPP design problem can also be formulated as a choice amongst pairs of primary-backup routes [48]. Each demand relation is provided with a set of eligible disjoint route pairs (one primary and one backup) to select from. Under the same objective of minimizing working and spare capacity costs, the model determines which pair of routes each demand relation is assigned for primary and backup routing, subject to the same sets of constraints above. In other words, there is only a single primary and backup route per demand relation, working capacity is equivalent to the primary routing, and spare capacity is sufficient to accommodate all simultaneously required backup routes for each span failure scenario. For more details on this design model, see [48] (pp. 417-420).

One final option is to formulate the SBPP model in a manner similar to the path restoration model that follows in Section 5.1.3. This would require that the restoration flow assignment variables in the path restoration model be failure independent (by addition of an equality-type constraint or simply redefining them to remove any failure scenario associations) and the stub release decision variables (or parameters as the case may be) be set to zero. This new model would not, however, restrict each demand to use of only a single primary and backup route pair, but rather would allow multiple primary and backup routes per demand relation (subject to integrality considerations, of course). Since the binary variables of the original SBPP model would effectively be replaced with integer variables, the new model is also easier to solve.

5.1.3 PATH RESTORATION

As mentioned in Section 4.4.3, path restoration employs a multi-commodity maximum flow (MCMF) type of simultaneous end-to-end rerouting over the available restoration routes. The primary difference between path restoration and SBPP is that in path restoration, there is no single pre-determined restoration route for each working path, but rather a collectively re-organized end-to-end rerouting of affected working paths in a manner that is specific to the failure scenario. Stub release further complicates the formulation of the ILP model for path-restorable network design (stub release is not applicable in the span restoration and SBPP models).

We begin with the path-restorable SCA network design problem, and we add to the notation already used for previous design models.

New Sets:

- P^r is the set of all distinct eligible routes available to carry restoration flow for demand relation *r*, and is typically indexed by *p*.
- $D_i \subseteq D$ is the set of working lightpath demands whose working route crosses span *i*, and is typically indexed by *r*. Note that here we assume that the working routes are pre-determined (hence this model is SCA), and any given demand relation is routed completely over a single working route. If for efficiency or other reasons, demand from a single demand relation should be split amongst multiple working routes, then that demand relation can be expressed as multiple demand relations in a manner similar to that described in Section 5.1.2.

New Input Parameters:

- $\delta_j^{r,p} \in \{0,1\}$ is a parameter that encodes restoration routes. If $\delta_j^{r,p} = 1$, restoration route *p* used for restoration of demand relation *r* crosses span *j*. If $\delta_j^{r,p} = 0$, restoration route *p* used for restoration of demand relation *r* does not cross span *j*.

New Decision Variables:

- $f_{i,p}^{r} \geq 0$ is the amount of restoration flow assigned to restoration route p for restoration of demand relation r in the event of failure of span i.
- $s_{i,j}^{*} \geq 0$ is the amount of stub-release capacity on span j that is released in the event of failure of span i.

The path-restorable SCA network design model is expressed as follows:

$$\text{Minimize} \quad \sum_{\forall j \in S} c_j \cdot s_j \tag{5.20}$$

$$\text{Subject to:} \quad \sum_{\forall p \in P^r | i \notin S_p} f_{i,p}^{r} = d^r \qquad \forall i \in S \quad \forall r \in D_i \tag{5.21}$$

$$s_{i,j}^{*} = \sum_{\forall r \in D_i | r \in D_j} d^r \qquad \forall i, j \in S | i \neq j \tag{5.22}$$

$$s_j + s_{i,j}^{*} \geq \sum_{\forall r \in D_i} \sum_{\forall p \in P^r | j \in S_p} f_{i,p}^{r} \qquad \forall i, j \in S | i \neq j \tag{5.23}$$

The objective function (5.20) minimizes the total cost of spare capacity needed to provide full restoration of all demands (for single span failures), and is the same objective function as the one in equation (5.1) for the span-restorable SCA model. The constraints set in equation (5.21) ensures that for any span failure scenario and demand relation affected by that failure, there is equivalent restoration flow over all eligible restoration routes available for use by that demand relation that do not cross the failed span. In equation (5.22) the amount of stub-release capacity on span j that is released in the event of the failure of span i is equivalent to the total number of lightpath demands over all demand relations whose working routes cross both span i and span j. We note that for any given pair of spans i and j, $s_{i,j}^{*}$ is uniquely determined by the pre-determined working routing, and so all $s_{i,j}^{*}$ values could be expressed as pre-calculated input parameters rather than as decision variables in a constraint set. This equation is subtly different than, say, the $f_{i,p}^{r}$ variables in equation (5.21) because there, the $f_{i,p}^{r}$ are all within the summation, and so none of the individual variables can be uniquely defined, whereas in equation (5.22), each $s_{i,j}^{*}$ variable is individually and therefore uniquely calculated separately from all of the others, with one copy of equation (5.22) for each. Finally, equation (5.23) places sufficient spare capacity on each surviving span *j* to fully accommodate all restoration flows simultaneously routed over it for each failure scenario, with a credit given for any stub-release capacity on the span.

Like previous design problems, the SCA formulation for path-restorable network design can be easily modified to address the JCA design problem of joint working and spare capacity optimization. As before, we modify the objective function, we add several new constraints to ensure the routing of working demands and adequate working capacity to support them, and then also modify two existing constraint sets (those in equations (5.21) and (5.22)).

In order to jointly optimize working capacities in addition to spare capacity, we use the same objective formulation as the previous JCA problems:

$$\text{Minimize} \quad \sum_{\forall j \in S} c_j \cdot \left(s_j + w_j\right) \tag{5.24}$$

The full formulation of the path-restorable JCA design problem is as follows:

$$\text{Subject to:} \quad \sum_{\forall q \in Q^r} g^{r,q} = d^r \qquad \forall r \in D \tag{5.25}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r \mid j \in S_q} g^{r,q} = w_j \qquad \forall j \in S \tag{5.26}$$

$$\sum_{\forall p \in P^r \mid i \in S_p} f_{i,p}^r = \sum_{\forall q \in Q^r \mid i \in S_q} g^{r,q} \qquad \forall i \in S \quad \forall r \in D \tag{5.27}$$

$$\dot{s}_{i,j} = \sum_{\forall r \in D} \sum_{\forall q \in Q^r \mid i,j \in S_q} g^{r,q} \qquad \forall i,j \in S \mid i \neq j \tag{5.28}$$

$$s_j + \dot{s}_{i,j} \geq \sum_{\forall r \in D_i} \sum_{\forall p \in P_i^r \mid j \in S_p} f_{i,p}^r \qquad \forall i,j \in S \mid i \neq j \tag{5.29}$$

The constraint sets in equations (5.25) and (5.26) are exactly the same constraint sets in equations (5.5) and (5.6) from the ILP formulation of the span-restoration JCA design problem; they ensure sufficient working flows to carry all demands, and calculate the working capacity required to accommodate those working flows, respectively. Equation (5.27) ensures that for each span failure scenario and each demand relation, there is sufficient total restoration flow over all applicable eligible restoration routes to fully restore all of that demand relation's working flow that crosses the failed span (if any). This equation replaces equation (5.21), and has been modified to account for the potentially multiple working routes being used by each demand relation (in equation (5.21), we assumed that all working flow for any given demand relation is routed over a single pre-determined working route). Equation (5.28) calculates the stub release capacities, and is a form of equation (5.22) that has been modified in a similar manner as equation (5.21) was modified into equation (5.27). Note that here, calculation of the $s_{i,j}^{*}$ variables must be done within the ILP itself and cannot be expressed as pre-calculated input parameters as they could in the SCA version of the problem because now, their values are a function of the $g^{r,q}$ decision variables rather than only input parameters. Finally, equation (5.28) is exactly the same as equation (5.23), which places sufficient spare capacity on each surviving span to fully accommodate all restoration flows simultaneously routed over it for each failure scenario, with a credit given for any stub-release capacity on the span.

Despite its more complex nature, path-restorable design is actually much easier to solve than the SBPP design problem. This is because the ILP formulation of the SBPP problem requires the use of integer binary decision variables (x_r^b, y_r^p, and $z_r^{p,b}$), which cannot be relaxed; $x_r^b = 0.75$ for instance has no real meaning since a route can only be used ($x_r^b = 1$) or not ($x_r^b = 0$). This greater number of integer decision variables therefore requires the solution of a great deal more nodes on average in the branch-and-bound tree when solving the ILP problem. We should note, however, that solving the JCA version of the path-restorable network design problem is considerably more difficult (i.e., time consuming) than solving the SCA version of the problem, and both versions are more difficult than the span-restorable network design problem, which has significantly fewer constraint sets and decision variables.

5.1.4 *P*-CYCLES

The final mesh restoration mechanism for which a network design problem will be modelled herein in the form of an ILP formulation is *p*-cycle restoration. The *p*-cycle design problem differs structurally from the span-restorable, SBPP, and path-restorable design problems in one key fashion: the optimization is in the form of a choice between eligible cycles rather than a choice between eligible routes. As such, we need to introduce new notation to properly express cycles and to identify straddling spans in addition to on-cycle spans. In addition to some notation used in previous design models, we add the following new notation.

New Sets:

- C is the set of eligible cycles that can be used to provide *p*-cycle protection to working links in the network and is indexed by *p*.

New Input Parameters:

- $X_{p,j} \in \{0,1,2\}$ is a parameter equivalent to the number of protection relationships provided by a unit-sized copy of *p*-cycle *p* for working links on span *j*. Recalling from Section 4.4.4 that a *p*-cycle provides protection for one working link on an on-cycle span and two working links on a straddling span, $X_{p,j}=1$ if span *j* is an on-cycle span for cycle *p*, $X_{p,j}=2$ if span *j* is a straddling span for cycle *p*, $X_{p,j}=0$ if span *j* is neither an on-cycle or straddling span for cycle *p*.

New Decision Variables:

- $n_p \geq 0$ is the number of unit-sized copies of *p*-cycle *p* placed in the network.

The full formulation of the JCA *p*-cycle network design problem is as follows:

$$\text{Minimize} \quad \sum_{\forall j \in S} c_j \cdot (s_j + w_j) \tag{5.30}$$

$$\text{Subject to:} \quad \sum_{\forall q \in Q^r} g^{r,q} = d^r \qquad \forall r \in D \tag{5.31}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r | j \in S_q} g^{r,q} = w_j \qquad \forall j \in S \tag{5.32}$$

$$\sum_{\forall p \in C} X_{p,i} \cdot n_p \geq w_i \qquad \forall i \in S \tag{5.33}$$

$$\sum_{\forall p \in C | X_{p,j}=1} n_p = s_j \qquad \forall j \in S \tag{5.34}$$

Equations (5.30), (5.31), and (5.32) are identical to prior equations seen in span-restorable JCA network design and path-restorable JCA network design. Equation

(5.30) minimizes the cost of working and spare capacity required in the network, while equation (5.31) and (5.32) ensure sufficient working flows to carry all demands, and calculate the working capacity required to accommodate those working flows, respectively. The constraint set in equation (5.33) ensures that for the failure of any span *i*, the *p*-cycles placed in the network provide enough protection relationships to fully reroute all of the working capacity on the failed span. Note that if $X_{p,i} = 1$, then *p*-cycle *p* will provide protection for one of the working links on span *i* per copy of the *p*-cycle, if $X_{p,i} = 2$, then *p*-cycle *p* will provide protection for two of the working links on span *i* per copy of the *p*-cycle, and if $X_{p,i} = 0$, then *p*-cycle *p* will not provide any protection for working links on span *i*. Equation (5.34) determines the amount of spare capacity on each span *j* as a result of the *p*-cycles placed in equation (5.33) that cross that span.

The SCA version of the above problem can be easily modelled by simply removing the constraint sets in equation (5.31) and (5.32), turning the w_i decision variables into input parameters (with values determined by pre-processing for, say, shortest path routing of demands), and removing w_i from the objective function.

5.2 PRE-PROCESSING FOR ROUTE ENUMERATION

All of the formulations presented in Section 5.1 are generalized models that are not specific to any particular network. They are applicable to any network by using appropriate pre-processing methods to fully express the design problems with the proper sets of eligible routes and/or cycles, and other input parameters required. Exact methods vary, but in the work done for this thesis, we use a depth-first search process to enumerate eligible routes and/or cycles and encode them as sets of spans (e.g., $S_q \subseteq S$) or as binary parameters (e.g., $\delta_j^{r,p} \in \{0,1\}$). That and a related shortest path algorithm used for performing working path routing (when applicable) are built into custom-designed software to generate data files specific to each network. A data file is then combined with a generalized model of the corresponding ILP (see Section 6.3 for details) and solved to carry out a network design.

5.2.1 ELIGIBLE ROUTE HOP OR DISTANCE LIMITS

In order to obtain the strictly optimal solution to one of the design problems, the eligible route and/or cycle sets need to contain all distinct routes[8] and/or cycles. However, for even moderately sized networks, such route sets and cycle sets are very large. For instance, in one 15-node 30-span test network used herein (see Section 6.1), there are 190425 distinct routes and 3969 distinct cycles. For slightly larger networks, the number of eligible routes can easily exceed the capabilities of the computers we use to solve such problems, and simply expressing the problem may even be beyond the available memory limits.

A common and practical approach to dealing with this issue is to impose a *hop limit* (*H*) on the eligible routes, such that only those routes composed of *H* or fewer spans are considered as eligible routes. The thinking is that since using shorter routes will require less capacity than using longer routes, then providing the problem with only short routes should allow a relatively efficient design. However, it is clear that as *H* is increased, more sharing-efficient patterns of re-routing are permitted, as was demonstrated in [60]. This allows for a fairly good trade-off between problem complexity and solution quality. A large *H* provides a greater number of routing options for the problem to choose from thereby reducing overall design costs but increasing the complexity of the problem (the greater the number of eligible working and/or restoration routes, the greater the number of constraints and variables in a design problem). A smaller *H*, on the other hand, restricts the number of routing options available, and so, although the problem's complexity may be greatly reduced, it is forced to select slightly less efficient and therefore more costly routing arrangements. As also shown in [60], there is some *threshold hop limit*, H^*, at which the theoretical minimum of capacity requirements is reached. In other words, the solution obtained when using no hop limit at all is no better than that obtained when using a hop limit of H^*. We note that the exact value of H^* is dependent on a variety of factors, including the network topology, demands, and span costs, and cannot be pre-determined analytically. Other options for restricting the number of eligible routes is to use distance limits, cost limits, or even optical path loss limits, all of which will provide trade-offs similar to that provided by a strict hop limit.

---

[8] A route is distinct from another if they differ by at least one span (i.e., it crosses at least one span that the other route does not). By *all distinct routes*, we mean every route that can possibly be drawn in the network graph that is distinct from all of the others already drawn.

One practical problem comes when a network contains sparsely connected regions with long chains and/or low nodal degree as well as other more richly connected regions with higher nodal degree. In this case, a hop limit of $H = 10$, for example, may be necessary to provide even a single eligible restoration route for some spans in a sparsely connected region, while using that same hop limit might produce many thousands of eligible restoration routes in a richly connected region of the same network. We therefore use a related strategy that is both effective and practical at representing and solving the design models, and also greatly improves the scalability of the problem to permit solution of quite large network design problems with a variety of richly and sparsely connected regions. The idea is not to presume a specific network-wide hop or length limit and attempt to generate all distinct routes up to that limit, but rather to use a procedure that results in a specified number of the shortest distinct eligible routes at whatever hop or length limit is required to realize the specified number for each failure scenario (or demand relation in the case of working routes) independently of one another. So for instance, if we specify that we require at least 10 eligible restoration routes per failure scenario, then the procedure is to enumerate the 10 shortest distinct restoration routes for each failure scenario separately. For some such failure scenarios, these eligible routes may all be under three or four hops in length, while for another failure scenario in the same network, they may actually be 10 or 12 hops in length. Experience shows that in cases where the 10th shortest eligible route for a failure scenario is the same number of hops as the 11th shortest, and possibly also the 12th shortest and so on, then all equivalently short eligible routes could also be included with negligible impact on problem complexity.

5.3 HEURISTIC AND ALGORITHMIC APPROACHES

In addition to the optimal ILP models already presented, many non-LP sub-optimal *heuristics* for solving various network design problems have also been developed. A heuristic is an algorithmic sequence of rules or actions that specify a means to solve a problem in a finite number of steps. While good heuristics tend to be quite fast and easy to use, their main drawback is that they do not guarantee an optimal solution (or even a feasible solution). Most heuristics are very specific to the problem they have been designed to solve, and even when a solution is obtained, a heuristic generally has no way of specifying how far from the optimal solution its own solution is.

One of the earlier heuristics was the *Spare Link Placement Algorithm* (SLPA) [50], which provides a near-optimal assignment of spare capacity in a span-restorable mesh network. The first phase of SLPA, *forward synthesis*, is to start with a network design containing one spare link per span, and then iteratively improve on the design by performing various operations (either add a spare link, add a pair of links, or add a complete restoration path) such that restorability is improved at the lowest possible cost at each iteration. The second phase, *design tightening*, removes spare links by iteratively performing various operations (remove one spare link, remove two spare links and add one, or remove three spare links and add two) while retaining full restorability. Additional variations on SLPA are provided in [113]. The *Max-Latching SCP Heuristic* [55] also provides a near-optimal assignment of spare capacity in a span-restorable mesh network. It does so by iteratively determining lowest-cost restoration routes for each span one at a time. At each iteration, the costs of the restoration routes consider only additional spare capacity that needs to be placed above what has already been assigned for use by restoration routes for spans already made restorable in earlier iterations.

Heuristics are used to allocate spare capacity resources in an SBPP network in [14], [66], [77], [78], and [57]. The work in [14] discusses *Simulated Allocation*, which uses a randomized process of allocation and de-allocation of routes to develop an efficient design solution. [66] develops three different algorithms, including one that is effectively max-latching modified to incorporate an *adaptive weight function* and applied to SBPP. *Successive survivable routing* (SSR) in [77] and [78] is a matrix-based model that routes disjoint backup routes one at a time and updates backup paths to accommodate for currently routed demands and their backup paths. In [57], four versions of a distributed algorithm are used to calculate working and backup paths using various link weight metrics (least loaded, minimum hop count, minimum cumulative backup, and minimum interference routing algorithm).

Meta-heuristics are a broad class of very general heuristics that are applicable to a wide variety of different problems by modifying the underlying algorithmic methods contained within them. *Tabu search* and *simulated annealing* are two well-known and widely used meta-heuristics [86]. They are considered to be *neighbourhood search* meta-heuristics because they both start with some initial feasible solution and then search for better and better solutions by navigating (or performing *moves*) through a *neighbourhood* of other feasible solutions. A neighbourhood is a set of feasible solutions that can be reached from the current solution by way of a move, where a move is a simple operation that typically involves modifying the current solution by some incremental way, such as (in the case of network design) swapping one unit of flow from one eligible restoration route to another. While the overall intent is to continue making moves until the optimal solution (or at least a near-optimal solution) is obtained, neighbourhood search heuristics can easily become trapped within a local minimum within the feasible solution space. In order to prevent this, most such heuristics allow, or in some cases even encourage, *uphill moves*.

The overall approach of a tabu search heuristic is to ensure that a neighbourhood search doesn't become trapped within a repeating sequence of moves, and uses a *tabu list*, which records and disallows future visits to feasible solutions already visited. In a tabu search, uphill moves are allowed if it will help to avoid solutions already visited; otherwise, moves generally proceed towards improvements. Eventually, new regions of the feasible solution space are investigated while avoiding local minima. Tabu search was recently applied to ring network design in [86] and to SBPP network design in [14].

Simulated annealing is a similar meta-heuristic whose approach models the behaviour of metal annealing, where a metal is cooled and then reheated and allowed to re-cool to reorganize its crystalline structure in a more desirable fashion. The search proceeds through the solution space by not only accepting moves that improve on the solution but also accepting uphill moves with a probability that slowly diminishes as the heuristic progresses towards an optimal solution. For this reason, it is often referred to as a stochastic hill-climbing neighbourhood search heuristic. Simulated annealing heuristics were developed for mesh network design in [14] and [78].

A third meta-heuristic is a *genetic algorithm* [83], which is based on the biological concept of evolution within a population, and seeks to maximize a *fitness function* by producing multiple generations of a population through various evolutionary-type operations. Solutions are encoded in a structure that is easily represented as a set of chromosomes or genes, which make up the current population. In the case of network design, chromosomes could represent assignment of flows to eligible routes, working and/or spare capacity assignments, etc. Any given population contains a number of feasible solutions, each with their own genetic makeup. At each iteration in the genetic algorithm, the current population is transformed into a new one by performing cross-over and mutation operations on its members, where cross-over selects sections of chromosomes from two members of the population and a mutation is simply a random modification of a member of the population. Following the concept of survival of the fittest, only the best members of the new generation remain in the new population. The whole process is then repeated for several generations. A genetic algorithm was applied to a combined capacity and routing problem in [83].

While the above examples are quite varied in their approaches, it is by no means an exhaustive list of heuristics that have been applied to network design problems. However, for the most part, heuristics as described above are not the main focus of this thesis and except for where they apply directly to specific topics to be discussed later, we leave it to the reader to decide whether they are suitable for his/her particular purposes.

5.3.1 ILP-BASED HEURISTICS

While we often don't think of them as heuristic methods, one form of heuristic we use extensively in this thesis is the ILP-based heuristic. Like the heuristics discussed above, an ILP-based heuristic is a generally sub-optimal method of solving a problem. What differentiates this class of heuristic is that it utilizes some form of reduced-complexity ILP model within it, and may be no more complicated than directly solving a simplified ILP model of the full problem. So in the strictest sense, in the work presented in this thesis, we solve the design models discussed in Section 5.1 as ILP-based heuristics since we either relax integrality on some variables and/or provide only limited eligible route or cycle sets.

There are some heuristics, however, that are very algorithmic in nature, where the use of an ILP is merely one step in the process. Work in [114] for example iteratively solves a cutset-based ILP, starting with a very small number of cutsets considered, and repeatedly adds new cutsets as required to suitably constrain the problem without considering all possible cutsets. And in [89], an ILP-based heuristic is used for joint network topology and capacity optimization (see also CHAPTER 8 for other methods to solve this problem). This heuristic works by iteratively developing candidate network topologies using purely algorithmic methods and then passing them on to an ILP solver to perform working and spare capacity optimization.

CHAPTER 6
EXPERIMENTAL SET-UP AND BENCHMARKING[9]

6.1 NETWORK TOPOLOGY MODELS

The design methods developed and discussed in this thesis are validated and characterized through comparative design trials conducted on a suite of 163 test network topologies, which we divide into six groups or *families* of related networks, as in [25] and [54]. Each family is headed by a *master network* with an average nodal degree of 4.0. While these master networks do not represent real network topologies per se (they were created as test network topologies within our research group), they are characteristic of real transport network topologies in the sense that they are planar or nearly planar, their nodal connectivities exhibit a high locality (i.e., they generally only connect to other nodes near them in the plane of the graph), and their nodes have a range of nodal degrees typically not exceeding twice the average nodal degree of the network as a whole. Each other member of a family is obtained by applying a succession of individual span removals to the high-degree master network while keeping all nodal positions (and end-to-end demands) fixed. More precisely, starting with the master network, one span at a time is removed from the graph, with each span removal resulting in a new subgraph, which becomes another member of the network family. Span removals are done pseudo-randomly[10], subject to retaining bi-connectivity, and continue until there is no span remaining whose removal would not violate bi-connectivity. Sparse regions and even chains of degree-two nodes were allowed to arise spontaneously as the average nodal degree is lowered through the span removals.

This technique allows us to directly test our various design methods and provide comparative analyses on networks having a systematic progression from high to low average nodal degree, and will better facilitate characterization and understanding of the design methods as a network's connectivity varies. Without such network families, studies on completely unrelated test networks would simply produce a scatter-plot in the space of capacity versus average nodal degree, and other underlying topological effects could easily obscure any real effects of the network connectivity on the design method.

---

[9] This chapter contains some material previously published in [25] and [54].

[10] We say "pseudo-randomly" because the span removed at each step was selected by a human designer with no preconceived notions about how to guide the evolution of the network and with no conscious bias towards any particular span or region of the network.

All master networks are labelled so as to indicate the number of nodes and spans in their graph. Network 15n30s1, for instance, is a network with 15 nodes and 30 spans. Individual networks in each family are labelled according to the master network and the number of spans remaining in the topology. For example, 15n30s1-19s corresponds to the network in the family headed by the 15n30s1 master network that has 19 spans remaining. All six master networks (15n30s1, 20n40s1, 25n50s1, 30n60s1, 35n70s1, and 40n80s1) are shown in Figure 6.1.

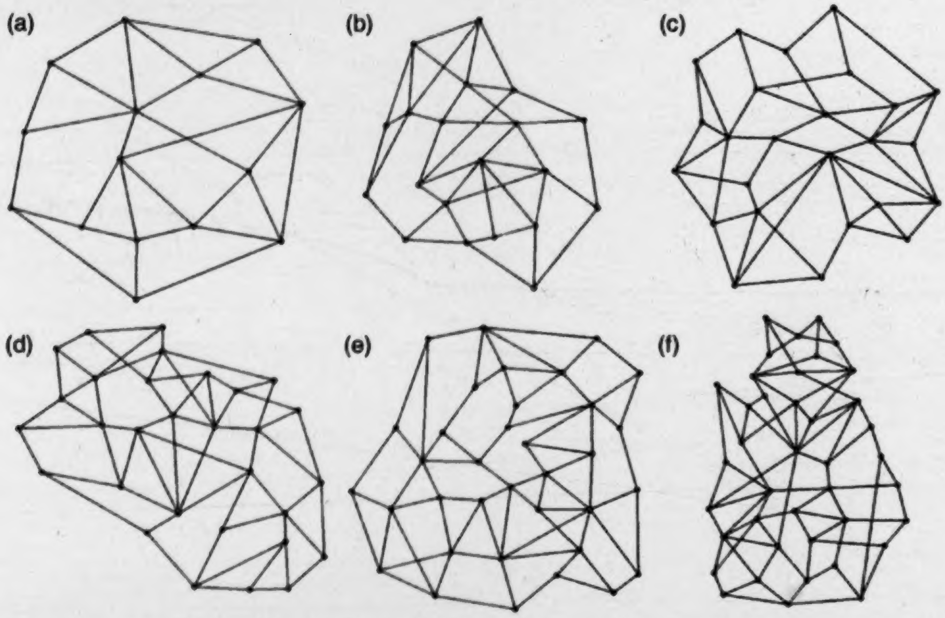


Figure 6.1 – Network topologies of the (a) 15n30s1, (b) 20n40s1, (c) 25n50s1, (d) 30n60s1, (e) 35n70s1, and (f) 40n80s1 master networks.

In all network topologies, the length of each span is the Euclidean distance on the plane between the end nodes the span connects. Unless specified otherwise, costs used by the various design problems in this thesis are equivalent to the span lengths, and so assuming length is measured in kilometres, resultant network costs can be thought of as corresponding to wavelength-kilometres. Topology files containing nodal coordinates, connectivity data, and span lengths and costs for all test case networks are provided in APPENDIX A.

6.2 DEMAND MODELS

As already discussed, optical transport networks may actually bear a mixture of IP/LAN, DS-n, STS-n, ATM, and leased whole-wavelength services. An important concept in the transport network is, however, that all such higher level service requirements, once forecast or otherwise assessed, can be viewed in aggregate and distilled down into a total demand for wavelengths between each O–D pair. Therefore, we do not need to deal with an array of traffic types, only a model for their aggregate totals between nodes. In real transport networks, lightpath demand matrices are a result of the myriad traffic types requiring service, and could follow a variety of models. In an attempt to keep the postulated wavelength demand matrices for test case networks as realistic as possible and reproduce plausible expectations about the real world, many different models can be defined. Following fairly common practice for generating instances of demand patterns for use in research studies, we typically employ an *inverse distance gravity model*, a simple *attraction model*, or a *uniform random model*.

In the gravity model, demands are generated from a mutual attraction effect proportional to node importance, but with an inverse distance dependency. In real networks, the population of a city or other regional measure of importance can be the basis of a node importance factor. Here, as a surrogate for measures such as population size or node importance, we use the degree of the node in each network. Using this model, the demand between a pair of nodes can be calculated as shown in equation (6.1), where $dem_{a,b}$ is the number of demand units exchanged between nodes a and b, d_a and d_b are the nodal degrees of nodes a and b, respectively, $dist_{a-b}$ is the distance between nodes a and b, and $Const$ is some constant scaling factor.

$$dem_{a,b} = \text{int}\left(\frac{d_a \cdot d_b}{dist_{a-b}} \cdot Const\right) \tag{6.1}$$

One strategy we often use is to set the scaling factor to be approximately half the average length of the spans in the network, implying that there is about a halving of the expected demand at one average span length. This model is often viewed as being a strongly localizing model of demand that may not be representative of some virtually distance-independent demands such as one might expect in the New York to Los Angeles (NY-LA) relation.

The attraction model is quite similar to the gravity model, with the main difference being the lack of inverse-distance effect (the distance term in equation (6.1) is set to 1.0). This allows generation of strong distance-independent demands such as the notional NY-LA example. It may also be more characteristic of a metropolitan-scale network where there is virtually no distance-based attenuation of demand, and of Internet-driven demand patterns where any given session or transaction is often as likely to be half-way around the world as it is to be in the same city. With distance no longer a factor, the constant now merely functions as a means of scaling the demands, and for comparative analyses, we often adjust the constant so that the average demand per O-D node pair approximates the model we wish to compare to.

In the uniform random model, every O-D pair is assigned a demand intensity from a discrete uniform random distribution. This model also has a basis in reality; demands in metro area networks and in quite densely populated areas, and increasingly also in the Internet, aren't necessarily related to either the degree (or some perceived importance) of the nodes, or the distance between them. In such a situation, a specified pair of nodes may be no more or less likely to exchange a large demand, as they are a small demand. This model is also often used to avoid any possible coupling with the tendency for high degree nodes (which get large demands under the other models) to also exhibit other properties such as being centrally located, or being high-degree anchors in low-degree regions of some networks. The uniform random model has no bias to any such effects and in comparative studies [54], doesn't appear to behave in any significantly different fashion than the other models. Unless specified otherwise, demands applied to the test networks in this thesis follow a discrete uniform random distribution in {1...10}, meaning each O-D node pair exchanges some uniformly randomly assigned number of demand units between one and ten, inclusive. In addition, all members of a given family use the same network demand matrix used for the family's master network. Network demand files containing the actual demands as generated by the uniform random model for each test network are provided in APPENDIX B.

A complicating trend in modern networks is the power-law type of behaviour often observed in Internet systems where, say, the hyperlink connectivity or popularity of web pages follows the power-law [29] or Zipf distribution [123]. With such Internet data traffic becoming an ever-increasing component of transport network demands, some type of power-law demand model might also be applicable. We can expect that future work in this area may make increasing use of such demand models. For the present work, however, we choose to make use of the uniform random model since, as mentioned above, it allows us to avoid any bias towards topological considerations in our demand data.

6.3 NETWORK DESIGN AND SOLUTION METHODS

All ILP design models used herein were implemented in AMPL Mathematical Programming Language version 9.0 [35] and solved using the CPLEX 9.0 MIP Solver [61] running on a 4-processor SUN UltraSparc III running at 900 MHz with 16 GB of RAM. Pre-processing for eligible working and restoration routes, eligible cycles, and other input parameters required by the ILP models was done on a dual-processor AMD Opteron 242 PC with 1 GB of RAM, running Windows 2000. Unless otherwise specified, all designs make use of the arc-path ILP models.

6.3.1 COMPUTATIONAL ASPECTS

A number of other aspects were common to all design types and their solutions. All working and spare capacity allocations were integer, corresponding to capacity design and restoration mechanisms at the wavelength level. For comparative studies we avoid any specific modularity assumptions [24], which could obscure the general underlying comparison of methods that is intended. Except where otherwise noted, results are based on a full CPLEX termination with a MIPGAP setting of 10^{-4} (i.e., solutions are guaranteed to be within 0.01% of optimal) for span-restorable and *p*-cycle network designs, and a MIPGAP setting of 10^{-2} (i.e., within 1% of optimal) for path-restorable and SBPP network designs.

Since we make use of arc-path ILP models, pre-processing by custom-designed software is required to enumerate the eligible working and restoration route sets and eligible cycle sets as appropriate. For span-restorable network designs, pre-processing for eligible working and/or restoration routes is almost trivial; the 5 shortest routes possible between the O-D node pair of each demand relation and the 10 shortest routes between the end-nodes of each span as specified by depth-first search are provided as eligible working and restoration routes, respectively. In cases where there is not sufficient diversity in the network to enumerate 5 distinct eligible working routes and 10 distinct eligible restoration routes, as many such routes as exist in the network are provided. Pre-processing for *p*-cycle network designs is no more complicated; the 5 shortest routes possible between the O-D node pair of each demand relation and the 1000 shortest possible cycles in the entire network are provided as eligible working routes and cycles, respectively. Again, if there does not exist 1000 distinct cycles in a network, then all possible cycles in the network are provided.

Pre-processing for path-restorable network designs is somewhat more complicated, however. Eligible working routes are enumerated as above (the 5 shortest per demand relation), but restoration routes are not so easy to enumerate, particularly for JCA designs, because failure scenarios are more complex. Recall that in path restoration, restoration routes are specific to the span that has failed, the working route used, and the demand relation affected, and so to be consistent with the other design models, we strive to enumerate 10 eligible restoration routes *per failure scenario*. The method we used is to provide the *n* shortest routes between the O-D node pair of each demand relation, where *n* is such that for each span on each eligible working route for the demand relation, there are at least 10 eligible restoration routes not crossing that span (*n* taking on a different value for each demand relation). While this may result in more than the required 10 eligible routes for some failure scenarios, it ensures sufficient restoration routes for all. If 10 eligible restoration routes do not exist for some failure scenario, or 5 eligible working routes do not exist for some demand relation, then all such possible routes are provided to the model.

Pre-processing for SBPP network designs was also less than straightforward, particularly so for the SCA designs where, as defined, primary working routing is via shortest routes. In some cases, it can occur that the shortest route completely bisects the network, with one of the route's end-nodes in one part of the network, and the other end-node in the other part of the network. We often call this the *routing infeasibility problem* or, alternatively, the *disjoint path infeasibility problem*. As demonstrated in Figure 6.2, if the primary working route between an O-D pair of nodes for a demand relation bisects the network (shown by black spans), then it becomes impossible to draw a backup route between that same O-D pair of nodes that is completely disjoint from the primary route.

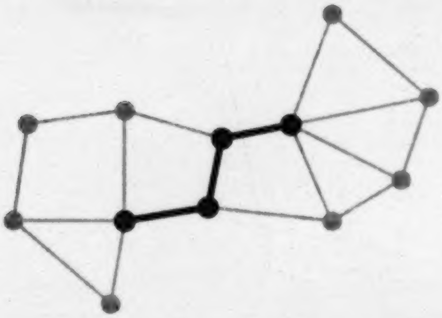


Figure 6.2 – Demonstrating the routing infeasibility problem in SBPP networks.

One solution to this problem is to simply identify when shortest path routing of primary working routes would result in such a routing infeasibility, and then find and use an alternative working route (say, the next shortest one) where a routing infeasibility will not arise. In this case, we can no longer claim that working routing is via strictly shortest routes, but rather that it is via the shortest routes with no backup routing infeasibilities. Once appropriate working routes have been determined, we then enumerate at least 10 eligible backup routes for each demand relation. The SBPP JCA designs, however, are more difficult to solve for reasons discussed in CHAPTER 5, and so we provide only 2 routes per demand relation as eligible primary routes, and at least the 10 shortest disjoint routes per demand relation as eligible backup routes. When enumerating the eligible backup routes, we take care to ensure that there are at least 10 such routes that are disjoint from each of the primary routes, so in effect, we quite often will have more than 10 eligible backup routes per demand relation. If there simply doesn't exist a backup route that is disjoint from one of the primary routes, then for that demand relation, we find the shortest cycle connecting its end-nodes and let the primary route be the shortest path around the cycle. The backup route is then the other path around the cycle, and if they exist, we also add up to 9 more routes disjoint from the primary route so as to enumerate 10 eligible backup routes as needed.

In all cases (except for some SBPP designs as noted above), working routing in SCA designs is via shortest routes, with working flow split evenly over equivalently short working routes, while retaining integrality of working flows.

6.3.2 VALIDATION CONSIDERATIONS

Prior to performing the tests and analyses presented in this thesis, we used several methods to validate all of our design models to ensure that they are, in fact, operating as intended. Our primary approach was typically to use very small test case network topologies, only a few demands, and a very limited set of eligible working and restoration routes and/or cycles. In some cases, this allowed us to solve the entire problem manually or at least to validate that, say, working and restoration routes and the associated capacity assignments are valid within the survivability mechanism modelled. We also inspected working and restoration route allocations in various larger test case networks to verify full restorability of the resultant designs. Where applicable, we also sometimes used special cases or otherwise checked that fundamental truths about a particular mechanism are not violated. For instance, we know that a span-restorable network cannot possibly achieve redundancy levels below the $1/(\bar{d}-1)$ lower bound, or that a path-restorable design with stub release must be at least as efficient as the corresponding design without stub release. Finally, the use of complimentary design methods acts as a final check on our results. An arc-path ILP model, for example, should produce designs with similar capacity requirements as a corresponding transhipment model (though not exact since, as we known, the arc-path model is limited by the set of eligible routes provided to it). For some restoration mechanisms, algorithmic design models were also available and could be used as further validation of our ILP design models.

6.4 COMPARATIVE ASPECTS OF BASIC MODEL BENCHMARKS

With such a wide variety of mesh restoration and protection mechanisms (and design models) available, it would be constructive to have quantitative analyses by which to compare and contrast them, as in [25] and [54]. Here, we provide benchmark network design solutions for each basic model on each test network topology described above, and directly compare them in terms of capacity requirements and redundancy, in an effort to obtain meaningful assessments of their merits and drawbacks.

6.4.1 TOTAL CAPACITY COSTS

Figure 6.3 through Figure 6.8 show normalized total capacity costs of optimally designed (i.e., minimum cost) networks of the various families designed for 100% restoration with span restoration, *p*-cycle restoration, SBPP, path restoration, and 1+1 APS. Each figure provides data for a single network family. Each data point represents the total wavelength-kilometres of working and spare capacity required to route all demands and provide for full restoration in the optimal solution for the member of the family with the indicated average nodal degree ($\bar{d}$) using the specified survivability mechanism and optimization model (e.g., SCA or JCA). In each figure, capacity costs have been normalized to that of the lowest-cost design over all networks in the family and all survivability mechanisms and optimization models. Within each figure, data points have been organized into curves corresponding to the various survivability mechanisms and optimization models. Note that by its very nature, the 1+1 APS protection mechanism doesn't require optimization in the strictest sense, and each corresponding data point was obtained by routing each demand via the shortest path and the next shortest disjoint path so that the primary working routing will be equivalent to the SBPP SCA designs. In some cases when disjoint path infeasibilities exist under shortest path routing, the shortest disjoint path pair is used instead.

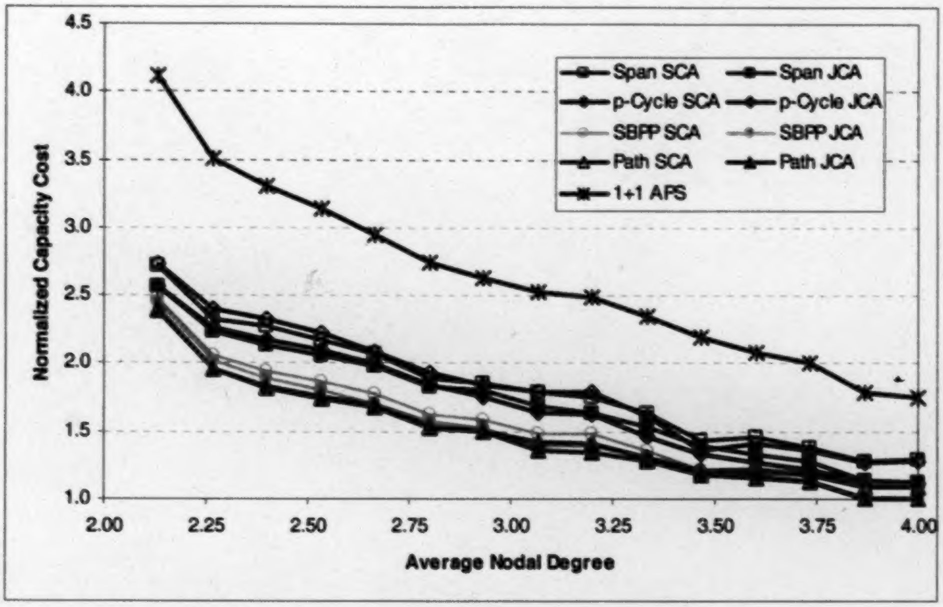


Figure 6.3 – Normalized total capacity costs for the 15n30s1 network family.

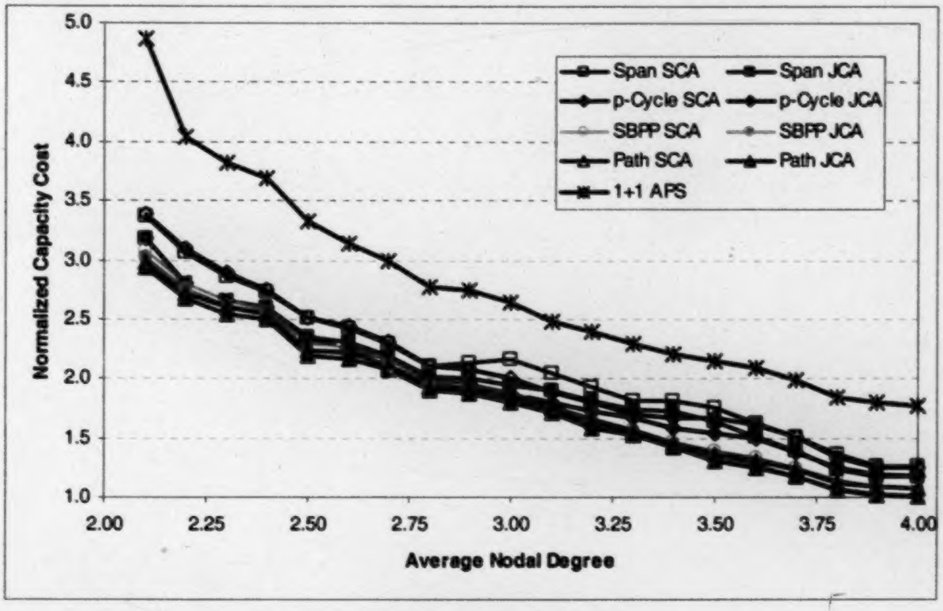


Figure 6.4 – Normalized total capacity costs for the 20n40s1 network family.

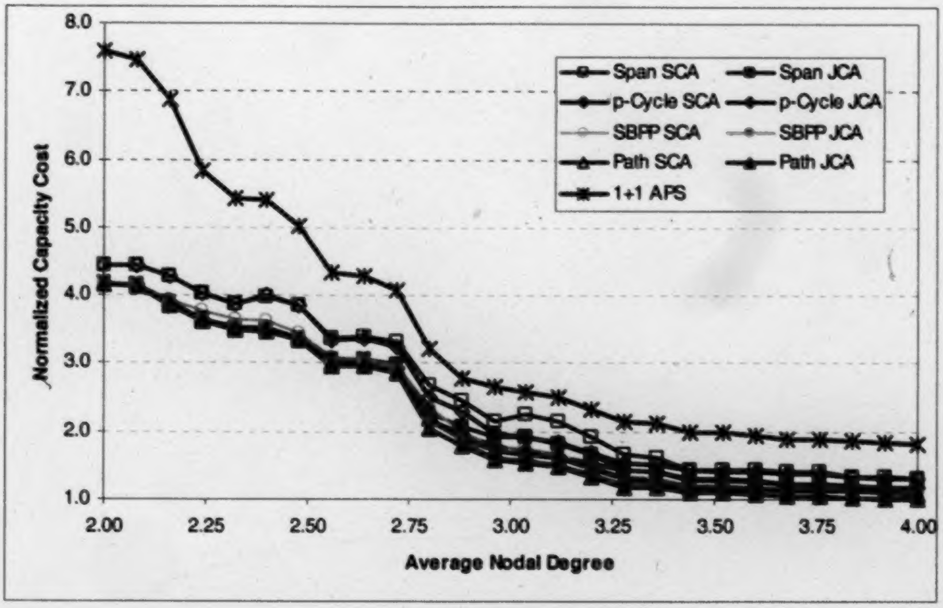


Figure 6.5 – Normalized total capacity costs for the 25n50s1 network family.

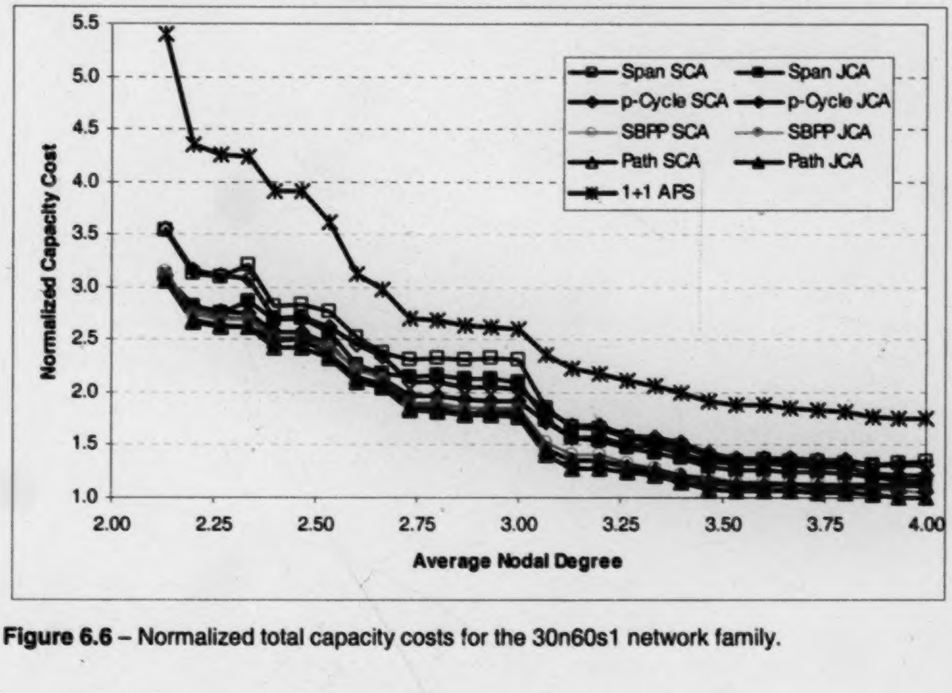


Figure 6.6 – Normalized total capacity costs for the 30n60s1 network family.

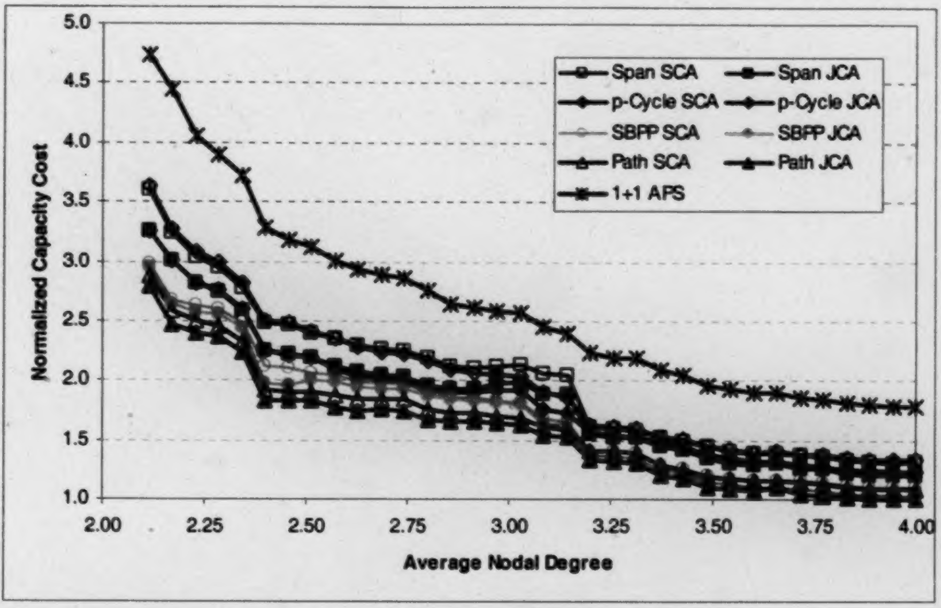


Figure 6.7 – Normalized total capacity costs for the 35n70s1 network family.

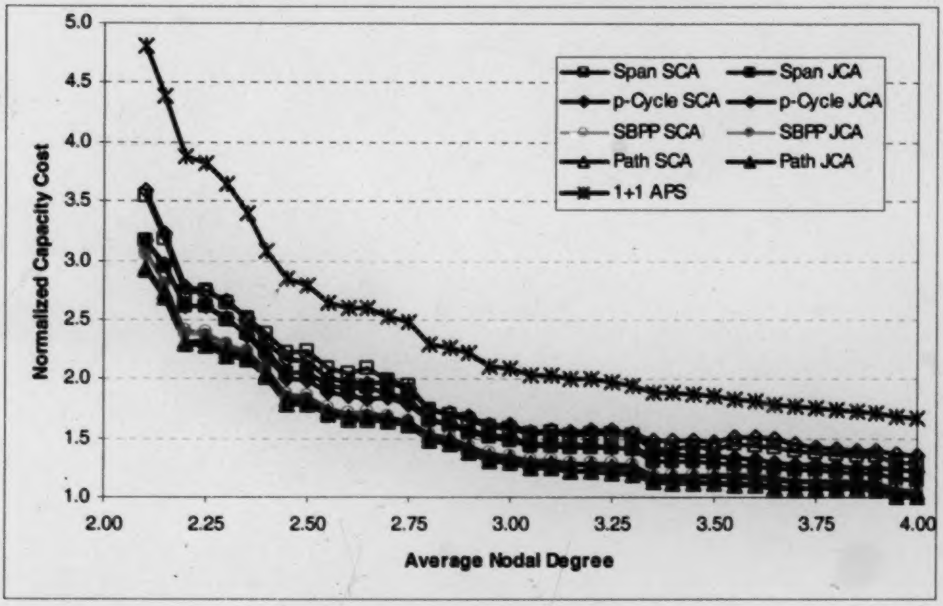


Figure 6.8 – Normalized total capacity costs for the 40n80s1 network family.

We can observe from Figure 6.3 through Figure 6.8 that the 1+1 APS network designs require significantly greater capacity than any of the other survivability mechanism, no matter the optimization model. On average, the 1+1 APS designs required 33% more total capacity that the next most costly mechanism (span restoration in most cases), and 68% more total capacity than path restoration JCA (86% more in the worst case).

In the slight majority of cases, span restoration SCA is the next most costly survivability mechanism, although for each network family, there are several portions of the curves where *p*-cycle SCA is more costly. Strictly speaking, the *p*-cycle restoration mechanism is actually a special case of the span restoration mechanism where restoration routes are formed between the end-nodes of the failed span but are restricted to following pre-formed cycles. Given sufficient eligible restoration routes, a span restoration SCA design will therefore always be feasible within the optimal *p*-cycle design, and in general will be able to achieve a less costly solution. So we would expect that the span restoration curves would all fall below the *p*-cycle curves. The reason this isn't always the case in our experiments here is due to the different eligible restoration route and cycle sets provided to the design problems. We recall from Section 6.3.1 that for the span-restorable designs, we provided each span failure with 10 eligible restoration routes, and for the *p*-cycle designs, we provided the problem with the 1000 shortest (i.e., least costly) eligible cycles. In the cases where the span-restorable designs require more capacity than the corresponding *p*-cycle designs, the eligible restoration route set in the span-restorable design problem is insufficient to find a feasible solution within the solution to the *p*-cycle design problem. Had we provided all possible restoration routes to the span-restorable design problems, however, those design solutions would be at least as low as the *p*-cycle designs, and in almost all cases lower.

Over all test cases, the total capacity costs for span restoration SCA and *p*-cycle SCA designs had an absolute difference of 2.6% on average, with the span-restorable designs being an average of 1.6% below the *p*-cycle curves. In one case (the 25n50s1-38s network), the *p*-cycle SCA curve was actually as much as 15.3% below the span restoration SCA curve, and it was never more than 3.3% above. The total capacity costs for span restoration JCA and *p*-cycle JCA designs also had an absolute difference of 2.6% on average, and the span-restorable designs were an average of 1.0% below the *p*-cycle curves. In the worst case (again, the 25n50s1-38s network), the *p*-cycle JCA curve was as much as 10.1% below the span restoration JCA curve, and it was never more than 5.3% above. Joint working and spare optimization provided an average of 8.2% reduction in total capacity costs for span-restorable network designs relative to SCA designs, and an average of 7.6% reduction in total capacity costs for *p*-cycle networks relative to SCA. For the 25n50s1 network family, JCA optimization provided an average total capacity cost reduction of 11.3% for span restoration (15.5% in the best case) and 9.9% for *p*-cycle restoration (14.2% in the best case). Joint optimization tended to provide the greatest benefits to sparsely connected networks. The reason for this may be related to the route enumeration method. In sparsely connected networks, finding a given number of distinct routes between the end-nodes of each span failure will enumerate a greater proportion of all possible routes within the network than will finding that same number in a network of high degree.

The SBPP SCA and JCA designs were the next lower, respectively, and for most test cases they were within just a few percent from each other. On average, the SBPP SCA curves were 5.4% below the *p*-cycle JCA curves and 12.6% below the *p*-cycle SCA curves, and the SBPP JCA designs were an additional 3.9% lower than that. In 20 of the 163 test cases (and 13 of them were from the 25n50s1 family), the *p*-cycle JCA total capacity costs were actually lower than those for SBPP SCA designs, though approximately half of those were within 1% (the SBPP designs' gap to optimality), and half of the remainder were within 2%. This is likely due to the benefits provided by jointly determining the working and restoration routing versus spare capacity optimization only, as well as the difference in eligible cycle and backup route sets provided to the two design problems. The *p*-cycle JCA designs were less costly that the SBPP JCA designs in 7 of the 163 test cases, but never more than 0.5%, which was less that the 1% gap to optimality of the SBPP designs.

Finally, as expected, the path restoration designs were the least costly of all. Over all of the test case networks, path restoration SCA designs were an average of 3.0%

less costly than SBPP SCA designs, and path JCA designs were 3.5% less costly than SBPP JCA designs. In both the SCA and JCA cases, path restoration tended to provide a greater benefit over SBPP in the more sparsely connected networks; for test cases with $\bar{d} < 3.0$, the path SCA designs were 4.1% better than the SBPP SCA designs and the path JCA designs were 4.2% less costly than SBPP JCA designs. On average, the path JCA designs required 4.4% less capacity than the path SCA designs, and in the most richly connected test cases the differences approached 10%. In fact, when considering only the test cases with $\bar{d} \geq 3.0$, the JCA designs were an average of 6.1% less costly, and path JCA costs of the 25n50s1 test cases with $\bar{d} \geq 3.0$ were an average of 8.0% less costly than the path SCA designs.

Another interesting finding is that the path restoration and SBPP designs benefited approximately half has much from joint optimization as the span restoration and *p*-cycle designs. The JCA designs for the two end-to-end survivability methods were an average of 4.1% less costly than the SCA designs, while the JCA solutions for the span restoration and *p*-cycle designs were an average of 7.9% less costly. This suggests that end-to-end survivability is less sensitive to working routing, perhaps because those survivability mechanisms are more easily able to take advantage of spare capacity sharing despite small inefficiencies in the working routing. In span restoration and *p*-cycles, routing working demands on slightly longer routes is able to provide reductions in spare capacity that greatly compensate for the increases in working capacity requirements, but this is less so in the end-to-end mechanisms.

There are several examples where the total capacity cost of a network appears to *increase* with average nodal degree. For instance, the span-restorable SCA design for the 20n40s1-28s network had a normalized cost of 2.104, the 20n40s1-29s network had a cost of 2.129, and the 20n40s1-30s cost 2.162. We know, however, that this should never occur since any design that is feasible in one network topology must also be feasible in any network topology containing all of the same spans as the first. So at the very worst, the 29-span and 30-span members of the 20n40s1 family should have the same design costs as the 28-span member. There are two reasons we see such increases here. The first is because of the eligible route enumeration methods employed. Recall that eligible route sets were generated separately for each individual test case in a network family, so each of the three aforementioned test cases will have different eligible routes available to them. Although the eligible restoration routes available for 20n40s1-29s are by definition at least as short (i.e., inexpensive) as those for 20n40s1-28s, they are not necessarily better collectively suited for spare capacity sharing. The second reason we see some higher design costs as $\bar{d}$ increases is that the working routes are shortest path routed, so the working capacities in 20n40s1-29s will be different (in this case less) than those in 20n40s1-28s. And as we've already seen, even slight differences in working capacities can have even greater effects, both positive and negative, on spare capacity costs. These abnormalities could have been repaired quite easily in the charts and calculations herein, but it was deemed more important to evaluate the various models where optimal designs for the individual test cases were allowed to arise in isolation from others in the same family since presumably, a network planner will typically be considering individual networks, not families of networks.

One final observation we make is that for the three larger network families, there appears to be a substantial drop in capacity cost at $\bar{d} \approx 3.0$. Similar drops were also seen at $\bar{d} \approx 3.0$ in two other test network families in [25] and [54]. In all cases, the drop in capacity cost occurs at the same $\bar{d}$ for all survivability mechanisms and design methods, although the drop appears to be more significant for span restoration and *p*-cycles and is much less perceptible for 1+1 APS. One possible explanation is that, particularly in span restoration and *p*-cycles, failures occurring adjacent to or near nodes of degree-2 are unable to make efficient use of shared spare capacity in their restoration (we recall that in span restoration and *p*-cycles, loop-back spare capacity is required on all spans adjacent to degree-2 nodes). So the $\bar{d}$ values at which the drops in the curves occur, likely corresponds to the topologies where most of the nodes in the network (or at least the key nodes) are of degree-3. One general implication of this is that network planners might wish to target topologies with $\bar{d} \geq 3.0$ and pay particular attention to ensuring that the key nodes (say, those centrally located and/or along high-bandwidth working routes) are at least degree-3.

6.4.2 SPARE CAPACITY COSTS

Analysis of spare capacity costs shows similar results to those above for total capacity costs, with the notable exception that here, differences between survivability mechanisms and design methods are enhanced. Figure 6.9 through Figure 6.14 show normalized *spare* capacity costs of optimally designed networks of the various families designed for 100% restoration with span restoration, *p*-cycle restoration, SBPP, path restoration, and 1+1 APS. Like the total capacity cost figures in the previous section, each figure provides data for a single network family. Each data point represents the total wavelength-kilometres of spare capacity required to provide for full restoration in the optimal solution for the member of the family with the indicated average nodal degree using the specified survivability mechanism and optimization model (e.g., SCA or JCA). In each figure, capacity costs have been normalized to the spare capacity cost of the $\overline{d} = 4.0$ master network using the path restoration JCA design method.

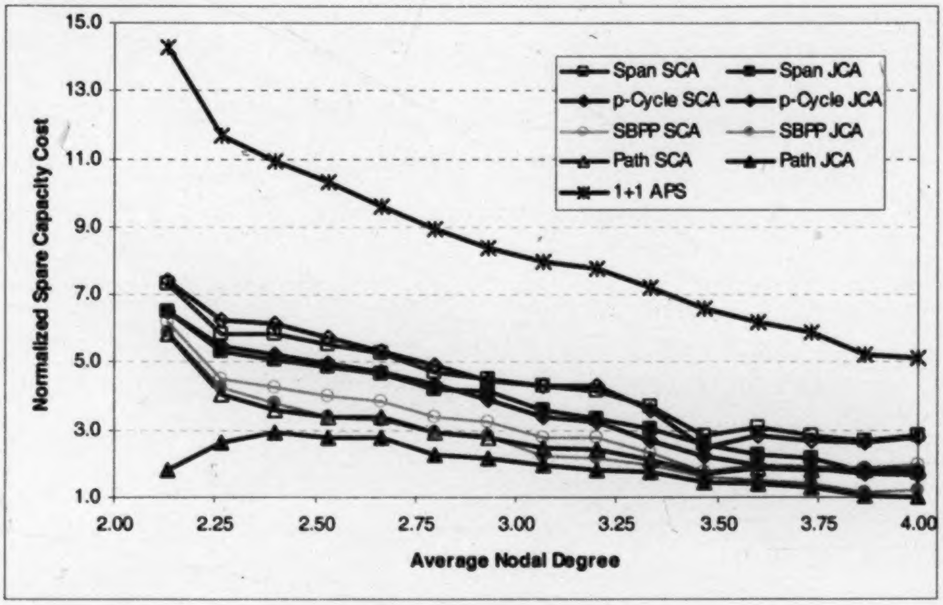


Figure 6.9 – Normalized spare capacity costs for the 15n30s1 network family.

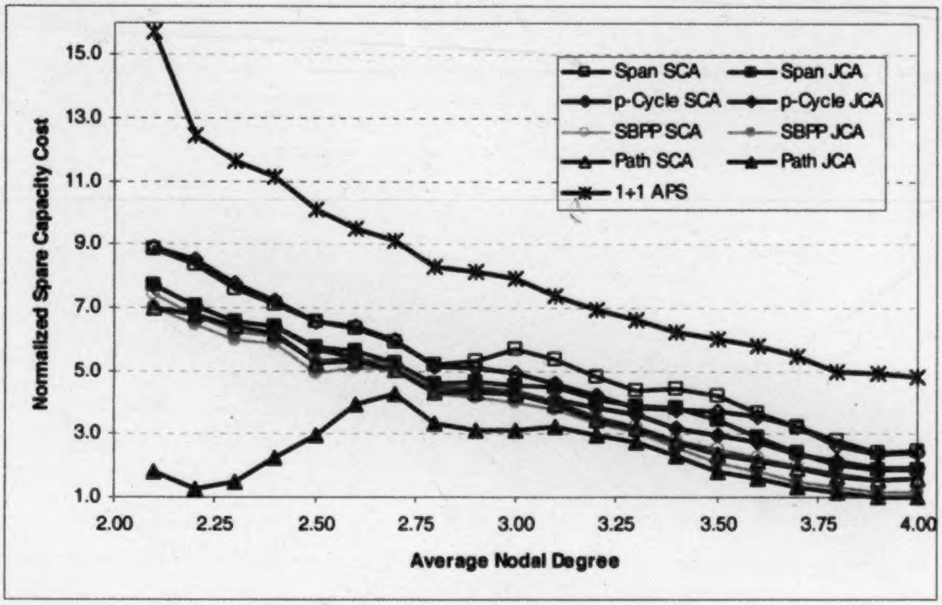


Figure 6.10 – Normalized spare capacity costs for the 20n40s1 network family.

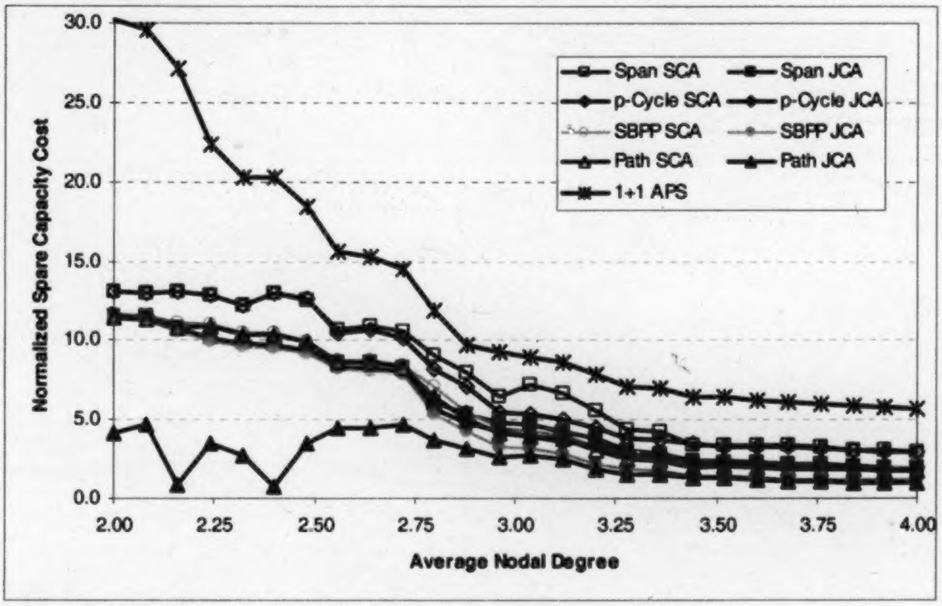


Figure 6.11 – Normalized spare capacity costs for the 25n50s1 network family.

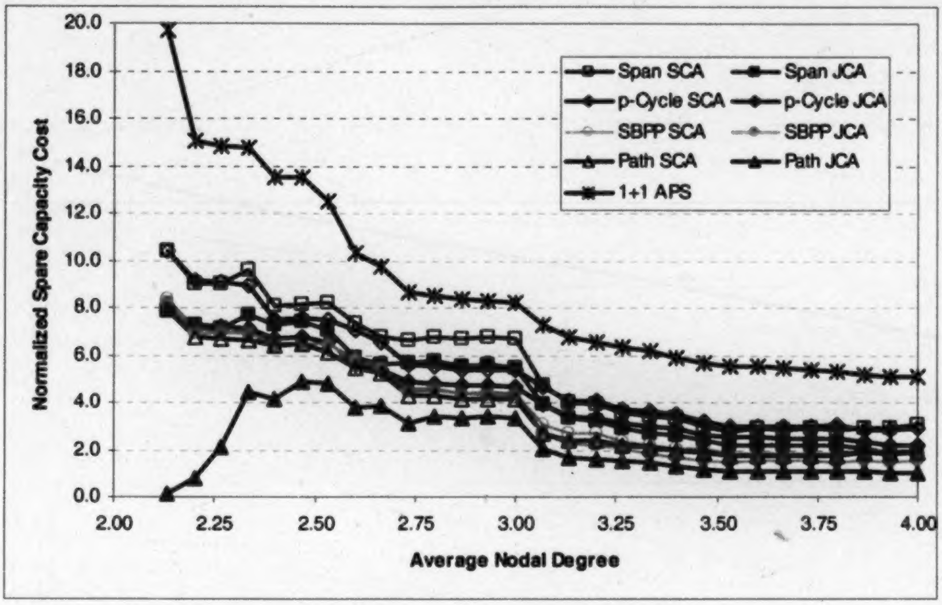


Figure 6.12 – Normalized spare capacity costs for the 30n60s1 network family.

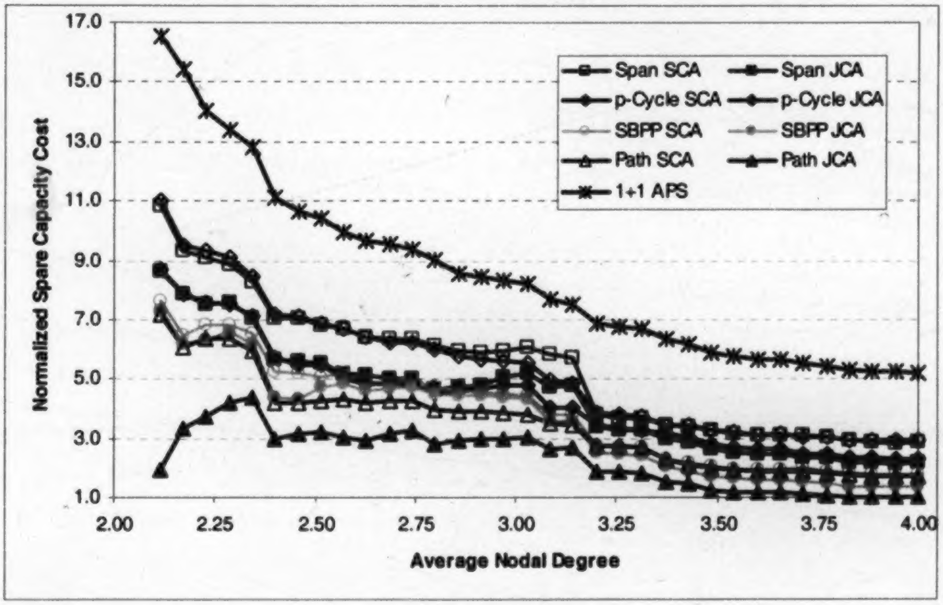


Figure 6.13 – Normalized spare capacity costs for the 35n70s1 network family.

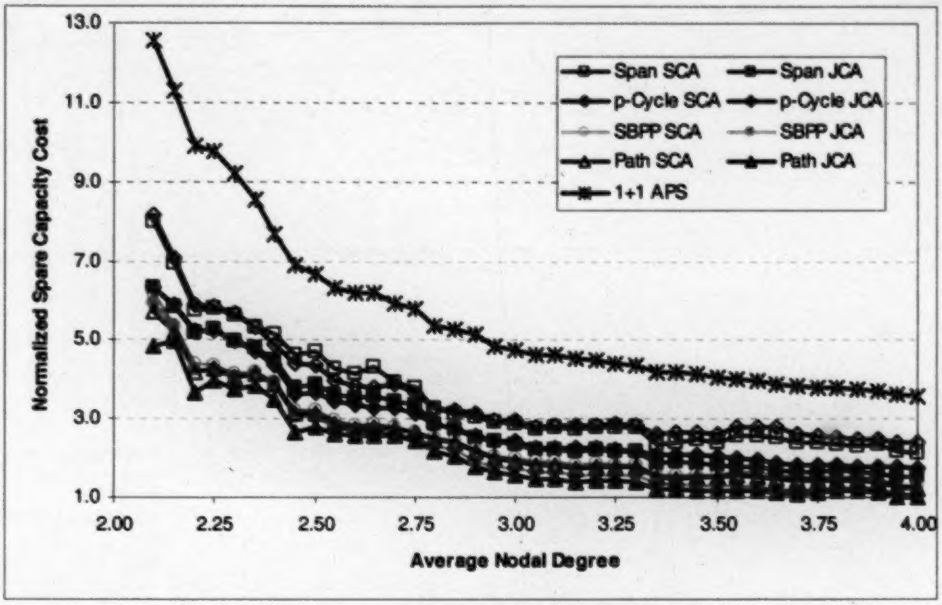


Figure 6.14 – Normalized spare capacity costs for the 40n80s1 network family.

We recall from Section 6.4.1 that 1+1 APS designs required 33% more total capacity than span-restorable or *p*-cycle SCA designs (the most costly of the two on a case-by-case basis) and 68% more than path-restorable JCA designs. In terms of *spare* capacity costs, however, 1+1 APS actually requires an average of 65% more than *p*-cycle restoration and 323% more than path restoration JCA (i.e., 4.23 times that of path restoration). In fact, over all 163 test cases, 1+1 APS designs never required less than double the spare capacity of path-restorable JCA designs (2.13 times to be specific), and it was quite common for a 1+1 APS design to required 5 or 6 times the spare capacity of the path-restorable JCA design. When using SCA optimization, span restoration required an average of 39.0% less spare capacity than 1+1 APS, *p*-cycle restoration required 2.9% less than that, SBPP needed 26.6% less than *p*-cycles, and path restoration needed 6.0% less than SBPP. Again we note that *p*-cycle designs were based on 1000 eligible cycles, while span restoration designs were provided 10 eligible routes per span failure. If we could provide the span restoration problems with all possible eligible restoration routes and the *p*-cycle problem with all possible eligible cycles, the order of their SCA spare capacity cost curves would be switched, with *p*-cycle designs costing at least as much as the span-restorable designs. There were also several cases where the SBPP SCA designs used slightly more spare capacity than path-restorable SCA designs, but this was never more than 2.5%, and in those cases, the total capacity costs used in the optimizations' objective functions were never higher in the path-restorable designs. For JCA optimization, span restoration required 59.4% less spare capacity than 1+1 APS, *p*-cycles needed 1.4% less than span restoration, SBPP needed 21.6% less than *p*-cycles, and path restoration required 30.3% less spare capacity than SBPP.

The benefits of jointly performing working and restoration routing is also most strongly evident when comparing spare capacity costs. In terms of *total* capacity, span-restorable and *p*-cycle JCA designs were 8.2% and 7.6% less costly, respectively, than span-restorable and *p*-cycle SCA. On the other hand, in terms of *spare* capacity, span-restorable JCA designs require 20.4% less than SCA designs and *p*-cycle JCA designs require 19.3% less than SCA designs. Joint optimization improved SBPP spare capacity costs by 14.3%.

It is in path restoration, however, that joint optimization provides the greatest reduction in spare capacity costs. On average, joint optimization provided a 37.2% reduction in spare capacity relative to SCA designs. In the most sparsely connected networks, spare capacity costs in JCA designs were often much less than half those of SCA designs, and on average, test case networks with $\bar{d} < 3.0$ required 41.9% less spare capacity. Even in the more highly connected networks, spare capacity reductions were quite significant – path restoration JCA designs required an average of 33.1% less spare capacity than path restoration SCA designs for networks with $\bar{d} \geq 3.0$. The path-restorable JCA design of one network (30n60s1-32s) actually required only 2.2% of the spare capacity required by the path-restorable SCA design, and there were two other test cases where the JCA design required less than 10% of the spare capacity of the SCA designs. These results were initially thought to be flawed and prompted a re-examination of the path restoration JCA ILP model for the source of error. However, the ILP models proved to be correct, and a thorough analysis of the resulting designs showed that the joint optimization allowed working routing to take advantage of stub release to an extent far beyond that of the SCA optimization. In fact, the path restoration JCA design of the 30n60s1-32s test case network required almost no spare capacity at all! However, as can be seen in Figure 6.6, the total capacity costs of the path restoration JCA design for the 30n60s1-32s network is virtually the same as the SCA design (they differ by 0.1%, which is well within the 1% gap to optimality for the design), because, as will be discussed in Section 6.4.4, demands are routed over significantly longer working routes in the JCA design than in the SCA design. Nevertheless, such a finding does suggest that for some networks, it might be possible to provide full restorability over stub release capacity and only a small amount of real spare capacity, while keeping working demands nearly shortest path routed. It appears that it is only in the most extremely sparse test case networks that such minute amounts of spare capacity is possible, and thorough analysis of the behaviour in such topologies is recommended for a better understanding of the effect.

One final note we make is that the drops observed in total capacity cost curves at $\bar{d} = 3.0$ are also evident here in the spare capacity cost curves, only more so, especially in the 30n60s1 and 35n70s1 network families.

6.4.3 WORKING CAPACITY COSTS

Figure 6.15 through Figure 6.20 show normalized *working* capacity costs of optimally designed networks of the various families designed for 100% restoration with span restoration, *p*-cycle restoration, SBPP, path restoration, and 1+1 APS. Like the total capacity cost figures in the previous sections, each figure provides data for a single network family. Each data point represents the total wavelength-kilometres of working capacity required to route all demands in the optimal solution for the member of the family with the indicated average nodal degree ($\bar{d}$) using the specified survivability mechanism and optimization model (e.g., SCA or JCA). In each figure, capacity costs have been normalized to the *spare* capacity cost of the $\bar{d} = 4.0$ master network using the path restoration JCA design method. The spare capacity cost was chosen for ease of comparison to figures showing normalized spare capacity costs in the previous section.

It is quite apparent from the figures that except for the path-restorable JCA designs, working capacity differs very little from one design method to the next. Obviously, the working capacity costs of the four SCA design curves, as well as the 1+1 APS curve will be identical since they all arise from the same shortest path routing algorithm. We note, however, that the working capacities for SBPP SCA and 1+1 APS aren't strictly identical to those of span restoration SCA, *p*-cycle SCA, and path restoration SCA. In a few cases, the working routes in the SBPP and 1+1 APS designs had to deviate from shortest paths to overcome disjoint path infeasibilities as discussed in Section 6.3.1. Over the 163 test cases, the SBPP SCA and 1+1 APS working capacities were an average of .78% higher than shortest paths. Over the 50 test case networks, where SBPP SCA and 1+1 APS working routing deviated from shortest path routing, they increased only 1.94%. We also observe that only 15 of those test cases had working capacity costs more than 2.5% higher than shortest paths.

It is also quite striking how joint optimizations for span restoration, *p*-cycles, and SBPP match so closely with the shortest path routing. Working capacity costs for the span-restorable JCA designs was an average of 4.3% higher than span-restorable SCA designs, while joint optimization increased working capacity costs by 4.0% in *p*-cycle networks and 2.6% in SBPP networks. Those slight increases in working capacity resulted in significantly larger decreases in spare capacity costs; 20.4% for span restoration, 19.3% for *p*-cycles, and 14.3% for SBPP (see Section 6.4.2). This observation, taken with discussions from Section 6.4.2, supports the thought that only a very small deviation in working routing is all that is needed for restoration and protection paths to find much more economical routes. It also confirms that restoration routing and subsequent spare capacity requirements are very sensitive to the specific working routes used.

Working capacity costs were the highest in path-restorable JCA designs, and were an average of 21.7% higher than path-restorable SCA working capacity costs over all 163 test case networks. For networks with $\bar{d} \le 3.0$, the effect was even greater; working capacity costs increased an average of 36.6% over shortest path routing. While a 21.7% increase in working capacity costs might seem quite high, we can refer to Section 6.4.2 where we saw a 37.2% decrease in spare capacity costs in the path-restorable JCA designs relative to path-restorable SCA, more than enough to offset the working capacity increase.

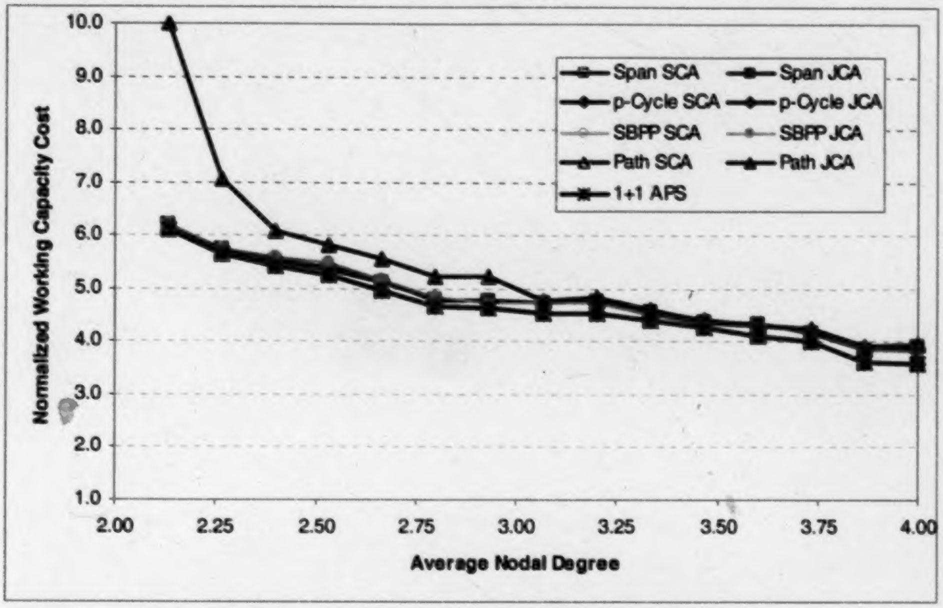


Figure 6.15 – Normalized working capacity costs for the 15n30s1 network family.

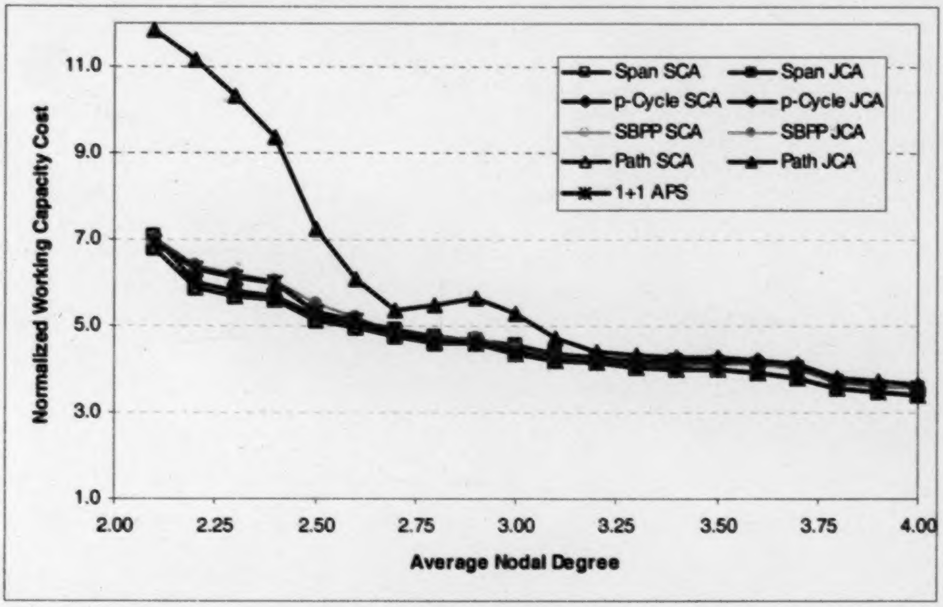


Figure 6.16 – Normalized working capacity costs for the 20n40s1 network family.

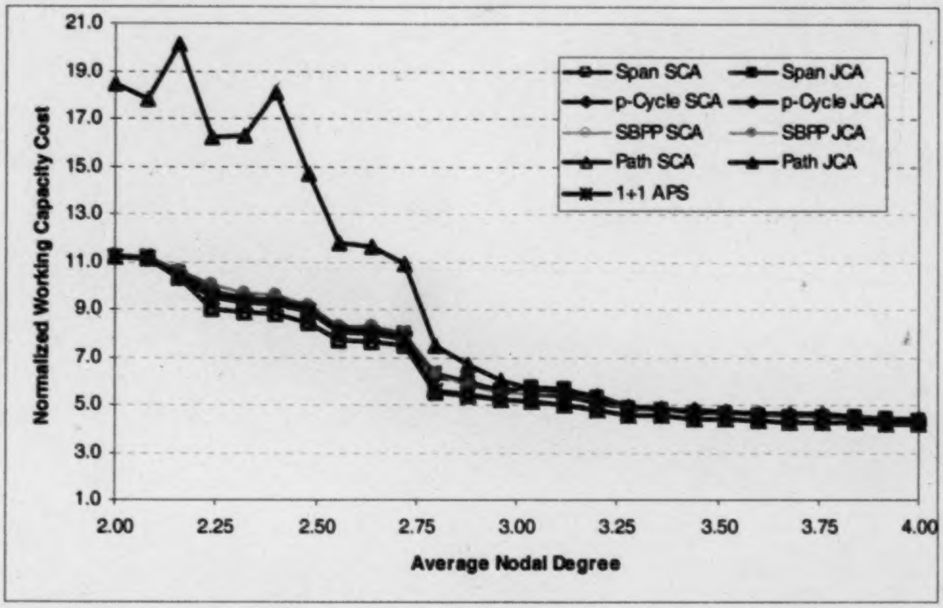


Figure 6.17 – Normalized working capacity costs for the 25n50s1 network family.

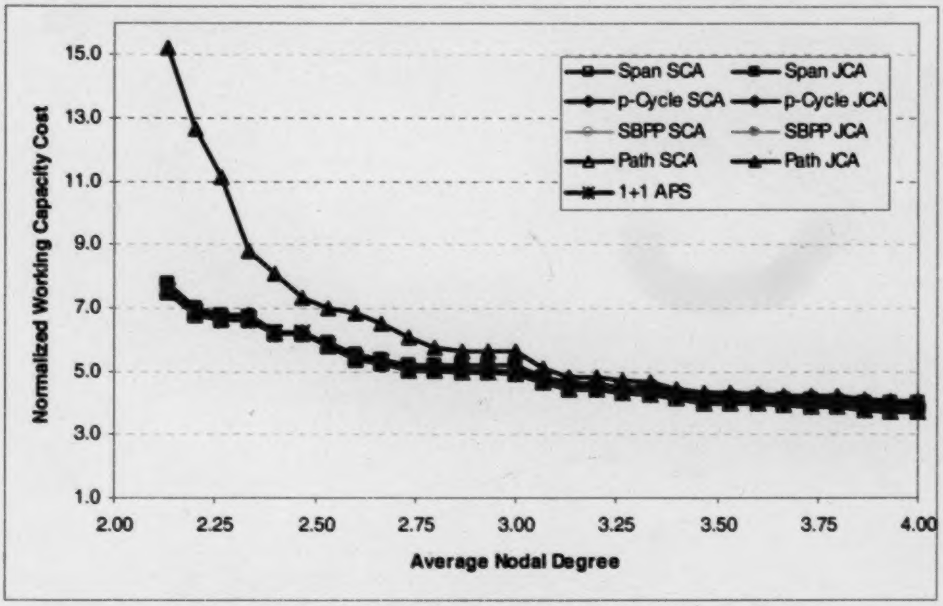


Figure 6.18 – Normalized working capacity costs for the 30n60s1 network family.

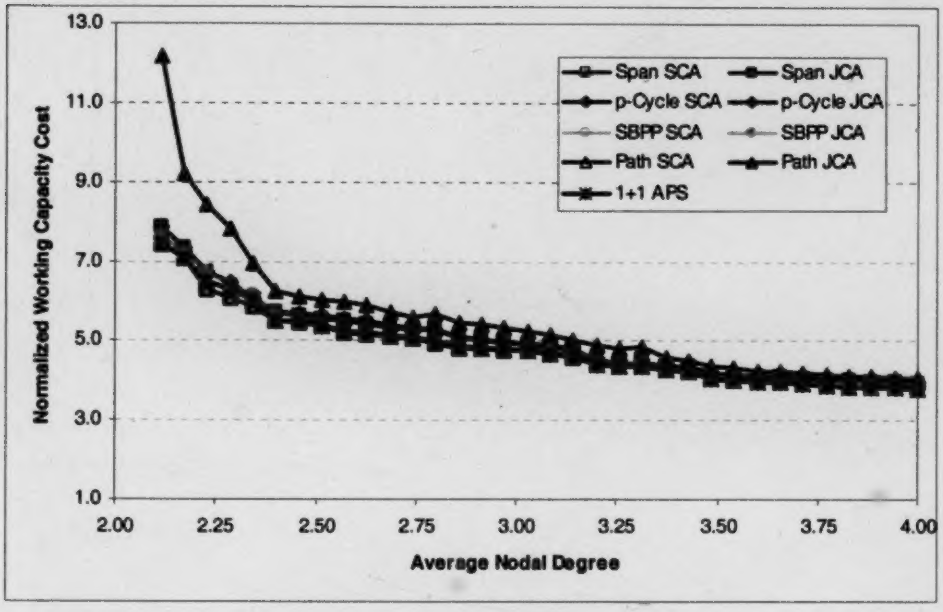


Figure 6.19 – Normalized working capacity costs for the 35n70s1 network family.

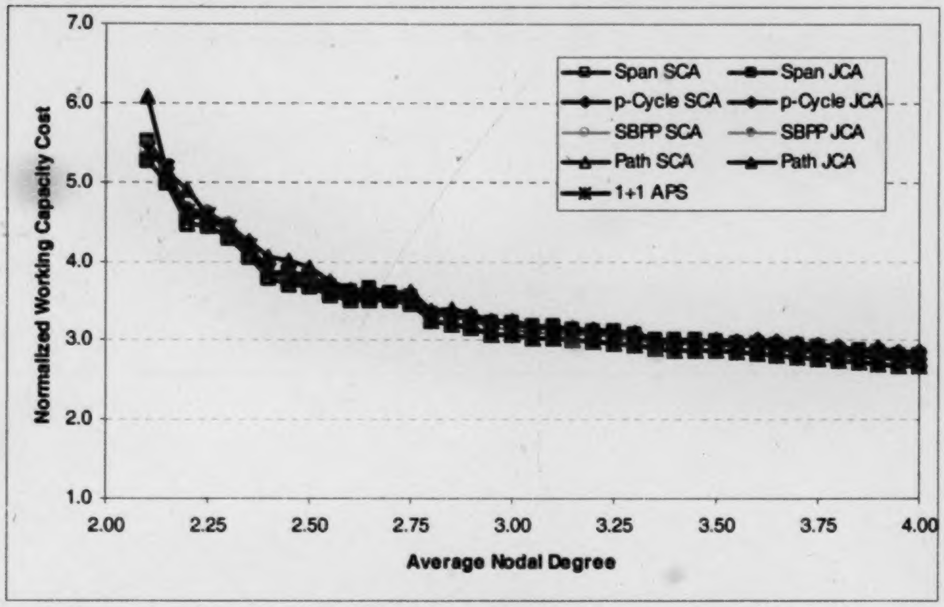


Figure 6.20 – Normalized working capacity costs for the 40n80s1 network family.

6.4.4 CAPACITY COST REDUNDANCIES

Figure 6.21 through Figure 6.26 show capacity cost redundancies of optimally designed networks of the various families designed for 100% restoration with span restoration, *p*-cycle restoration, SBPP, path restoration, and 1+1 APS. Like the capacity cost figures in the previous sections, each figure provides data for a single network family. Each data point represents the R_{cost} of the optimal solution for the member of the family with the indicated average nodal degree ($\bar{d}$) using the specified survivability mechanism and optimization model (e.g., SCA or JCA). The $1/(\bar{d}-1)$ lower bound on capacity redundancy in a span-restorable network has been added to each figure for reference and comparison purposes. While the $1/(\bar{d}-1)$ curve corresponds to unit capacity redundancy, R_{cap}, and is not strictly applicable for capacity cost redundancy calculations, experience shows that as long as span costs don't vary too greatly from span to span, $R_{cap} \approx R_{cost}$.

The first observation that we make is that as expected, the span restoration and *p*-cycle redundancy curves all fall above the $1/(\bar{d}-1)$ lower bound (remember that *p*-cycle restoration is a special case of span restoration so the lower bound applies to it as well). And since 1+1 APS is by definition at least 100% redundant, its redundancy is always significantly above the $1/(\bar{d}-1)$ lower bound, which is at most 100% redundant (at $\bar{d} = 2.0$). It is interesting to note that for most of the survivability mechanisms, redundancy is relatively flat over the range of average nodal degree up to $\bar{d} \approx 3.0$, which implies that when spare capacity increases as we move to lower $\bar{d}$, there is generally a proportional increase in working capacity. Then just as we observed in the total capacity and spare capacity cost curves, a drop in redundancy at $\bar{d} \approx 3.0$ is also evident here in most network families over most survivability mechanisms.

As seen previously in [25] and [46], SBPP and path restoration redundancies are both capable of reaching below the $1/(\bar{d}-1)$ lower bound since it applies only to span-restorable networks. For most network families and at most levels of connectivity, SBPP SCA, SBPP JCA, and path restoration SCA redundancies are still above $1/(\bar{d}-1)$. Path restoration JCA redundancies, on the other hand, are significantly lower, and for the most part, are at or below $1/(\bar{d}-1)$. In the 15n30s1 and 25n50s1 network families, path restoration JCA redundancy never exceeds 50%, it barely reaches 60% in the 35n70s1 network family, and only briefly tops 60% in a few individual test cases in the other two network families. In fact, for most network families, the maximum path restoration JCA redundancy in the worst case only marginally exceeds the redundancy of the span restoration, *p*-cycle, and SBPP SCA curves in the best case. For instance, in the 25n50s1 network family, the worst-case path restoration JCA redundancy is 49.7% at $\bar{d}=2.8$, while the best-case redundancies for the other mechanisms are 70.8% for span restoration SCA, 45.1% for span restoration JCA, 70.6% for *p*-cycle SCA, 45.4% for *p*-cycle JCA, and 46.2% for SBPP SCA.

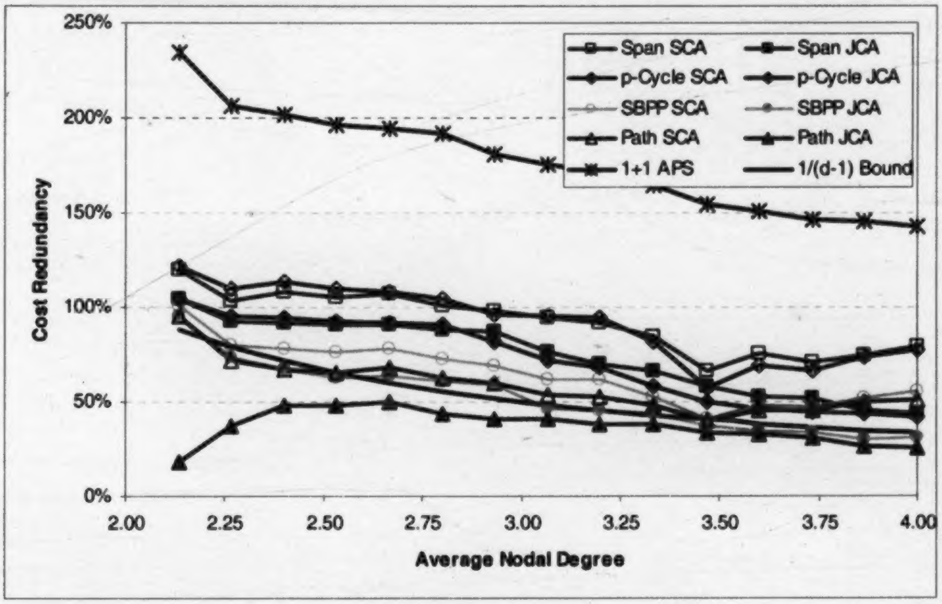


Figure 6.21 – Capacity cost redundancy for the 15n30s1 network family.

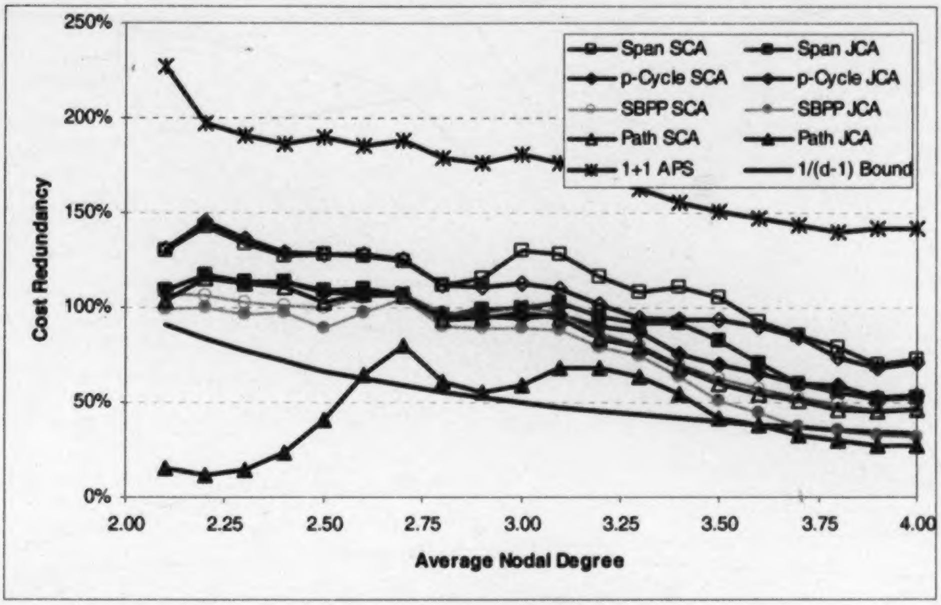


Figure 6.22 – Capacity cost redundancy for the 20n40s1 network family.

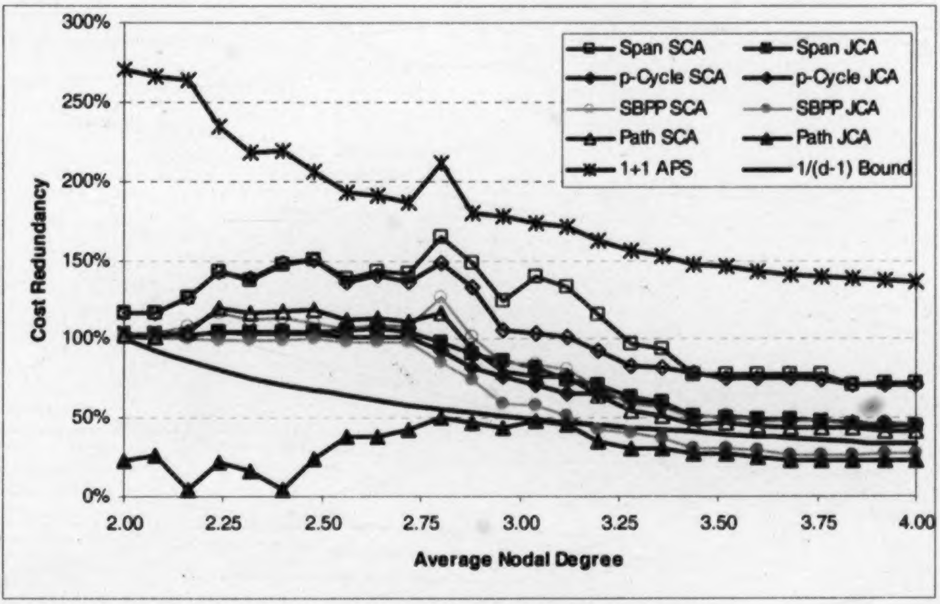


Figure 6.23 – Capacity cost redundancy for the 25n50s1 network family.

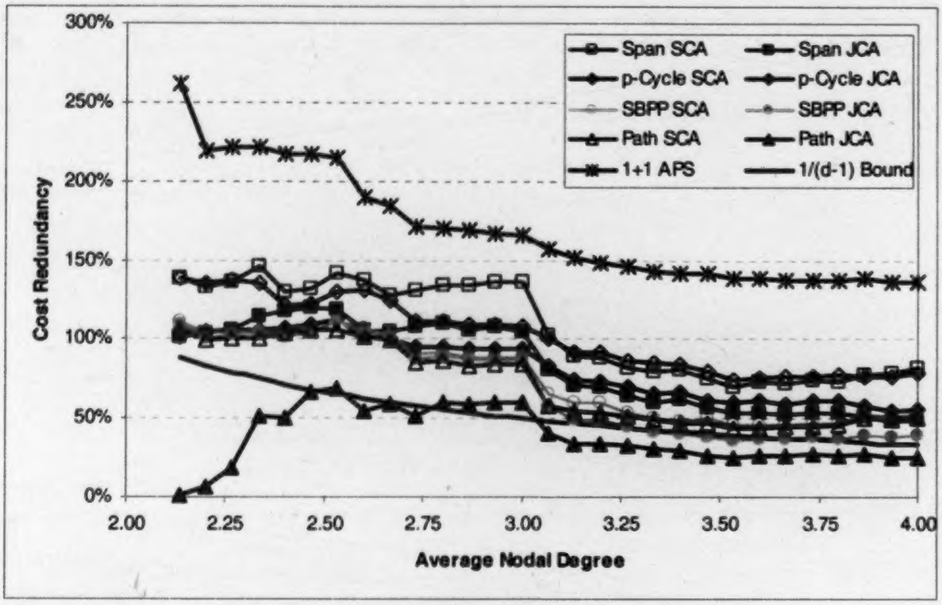


Figure 6.24 – Capacity cost redundancy for the 30n60s1 network family.

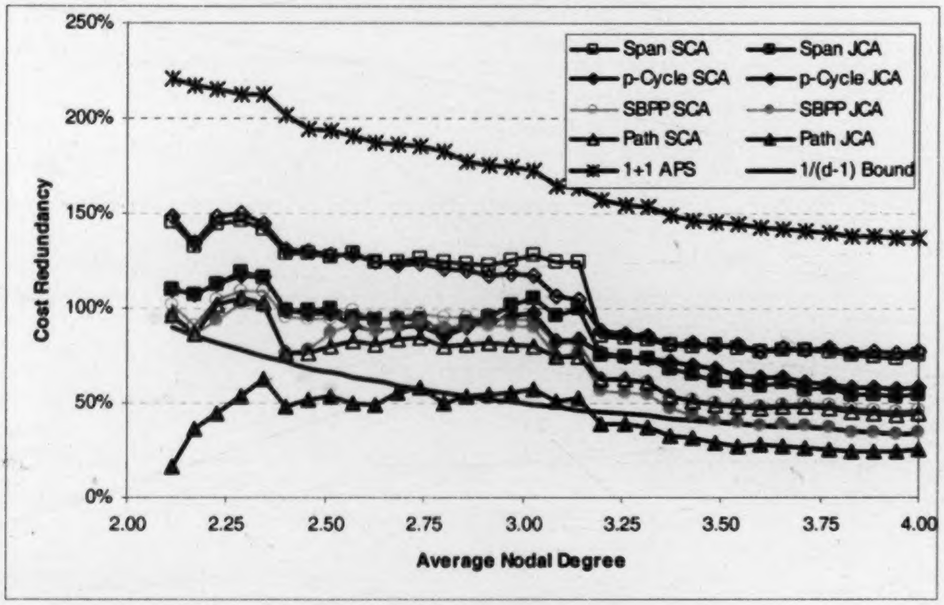


Figure 6.25 – Capacity cost redundancy for the 35n70s1 network family.

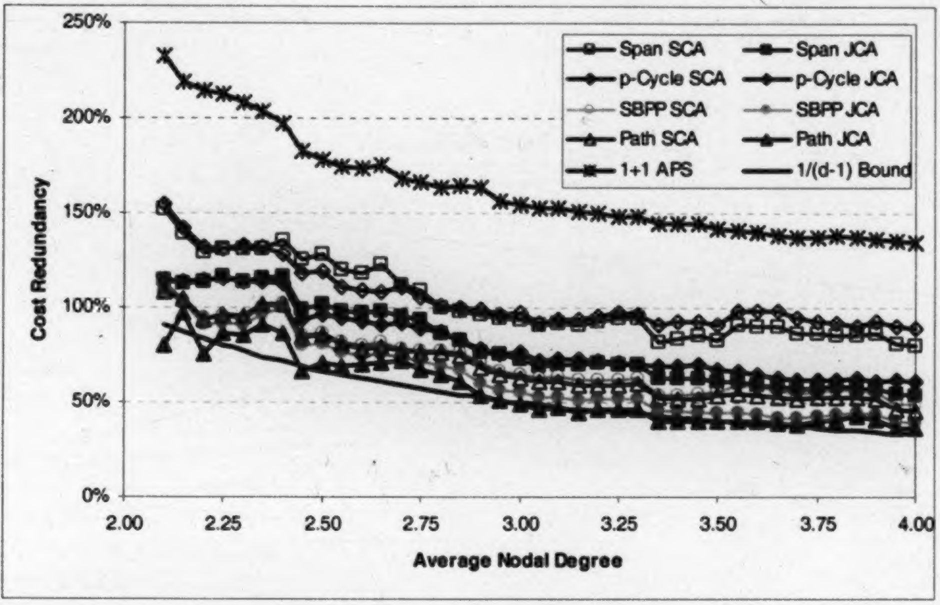


Figure 6.26 – Capacity cost redundancy for the 40n80s1 network family.

Perhaps the most remarkable observation we can make is that in the particularly sparse network topologies, path restoration JCA redundancies are so exceptionally small that in some cases they approach 0%. The 25n50s1-27s and 25n50s1-30s network topologies for instance, at $\bar{d} = 2.16$ and $\bar{d} = 2.4$, respectively, can achieve 4.5% redundancy, 30n60s1-33s can reach 6.6%, and seven other topologies of various $\bar{d} \le 2.32$ have redundancies ranging from 10% to 20%. Amazingly, the 30n60s1-32s network topology with $\bar{d} = 2.13$ can be designed with a path-restorable JCA redundancy of only 1.1%! To put that into perspective, the best span-restorable JCA redundancy out of all of the 163 test case networks was 43.3%. But recalling our earlier discussions on total capacity costs, we can easily understand that the low redundancies result primarily from exceedingly high working capacity costs and low spare capacity costs, which together sum to total capacity costs that are relatively in line with the other design methods, and which behave in a conventional manner (increasing as $\bar{d}$ decreases). As mentioned in Section 6.4.2, the knowledge that such low redundancies are even achievable provides meaningful insights into the benefits of stub release, and raises the possibility that extremely small amounts of spare capacity might be sufficient to fully protect some sparse networks if working lightpath demands can be appropriately attracted (say, by offering reduced fees).

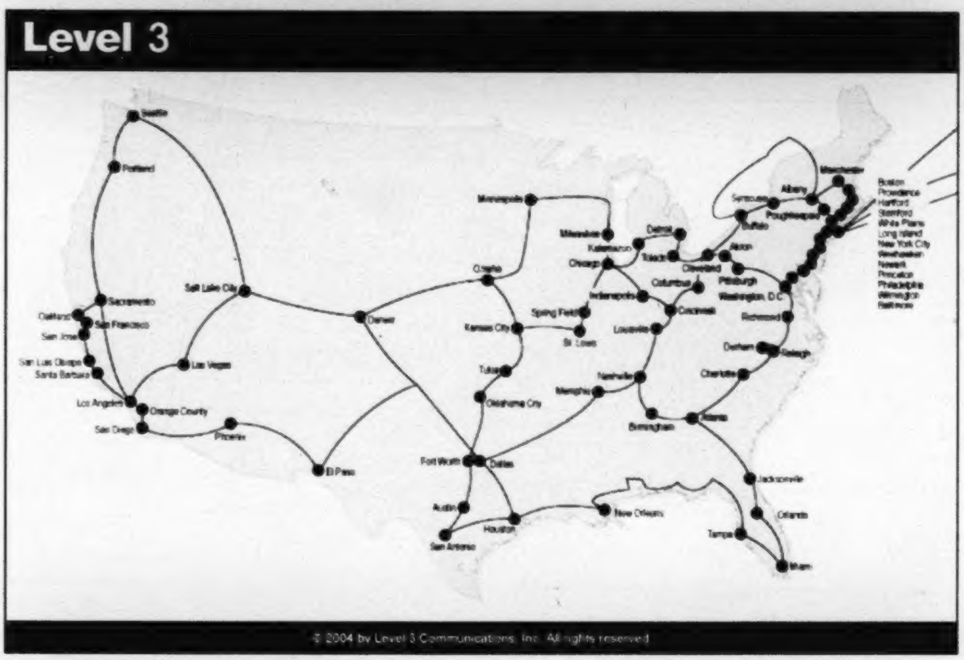
CHAPTER 7
META-MESH NETWORK DESIGN[11]

Recent advances in WDM-based switching technology are now giving rise to network elements (e.g., OXCs and OADMs) that are capable of manipulating individual lightwaves and provide true lightpath agility and reconfigurability in the transport network [87]. One advantage that these WDM networking elements offer is that they enable the concept of an *automatically switched transport network* (ASTN) [2], also called a *self-organizing* network [47]. But another advantage, and of significance to our present interest, is that these elements allow the use of mesh restoration schemes for the optical networking layer.

A key motivation for using mesh-based networks is the superior capacity efficiency attained through well-organized sharing of spare capacity resources. Among the various mesh restoration and protection mechanisms, span restoration is a particularly attractive choice because of its amenability to the PWCE concept (recall the discussion in Section 4.5), because of its simplicity and ease of implementation, and because of the self-organizing protocols available for it [45], [47]. We saw in CHAPTER 6 that span restoration and other restoration and protection mechanisms are particularly capacity efficient in richly connected network topologies (those with a high average nodal degree, $\bar{d}$), where a greater degree of sharing amongst restoration routes is allowed. Mesh restoration techniques are therefore well suited to European networks, which are typically of average nodal degree between 3 and 4.5. However, many North American networks are quite sparse in comparison. For instance, Level 3 Communications' USA backbone network, shown in Figure 7.1, has an average nodal degree of approximately 2.4 [75]. A very sparse graph can make the economic advantage of mesh-based networking questionable, and such sparse network topologies are often judged to be simply too low-degree to benefit enough from mesh restoration (relative to a ring-based status quo). After all, mesh efficiencies can only possibly occur at nodes with $d \geq 3$; a node with $d = 1$ is not restorable at all, and a node with $d = 2$ is already as well served by a shared-protection (BLSR-type) ring as it can be. Indeed, when the less costly OADMs used by rings (as opposed to the OXCs needed by mesh-based networks) are taken into consideration, this perception that the sparsest networks are better suited to rings may very well prove true in some cases, despite the lower capacity efficiency of rings; the lower cost of OADMs more than compensates for the increased capacity requirements.

---

[11] This chapter contains some material previously published in [52] and [54].


Level 3
Seattle
Portland
Sacramento
Oakland
San Francisco
San Jose
San Luis Obispo
Santa Barbara
Los Angeles
Orange County
San Diego
Phoenix
Las Vegas
Salt Lake City
Denver
El Paso
Omaha
Minneapolis
Milwaukee
Chicago
Kalamazoo
Detroit
Toledo
Cleveland
Akron
Pittsburgh
Columbus
Indianapolis
Cincinnati
Louisville
Spring Field
St. Louis
Kansas City
Tulsa
Oklahoma City
Memphis
Nashville
Birmingham
Atlanta
Fort Worth
Dallas
Austin
San Antonio
Houston
New Orleans
Jacksonville
Orlando
Tampa
Miami
Charlotte
Durham
Raleigh
Richmond
Washington, D.C.
Syracuse
Buffalo
Albany
Poughkeepsie
Manchester
Boston
Providence
Hartford
Stamford
White Plains
Long Island
New York City
Weehawken
Newark
Princeton
Philadelphia
Wilmington
Baltimore

© 2004 by Level 3 Communications, Inc. All rights reserved

Figure 7.1 – The Level 3 Communications, Inc. USA backbone network [75].

While the Level 3 network shown is but one example, due to geographic and demographic reasons, the reality is that many other North American *inter-exchange carrier* (IXC) networks necessarily have very sparse topologies as well ([1], [43], [118]). At least empirically it is well recognized that North American networks, especially in Canada and over large parts of the mid-USA, tend to be of lower degree than European networks. This is perhaps because there have been fewer revenue producing source/sink centres per unit geographic area to justify the historical development of a richer fabric of direct facility routes at the continental scale. And more recently, advances in transmission capacity, and related economy of scale in capacity-cost effects, only serve to reinforce the tendency towards sparse facility graphs [24]. With large amounts of capacity and economy-of-scale effects, it can often be economic to route longer distances over sparser graphs, rather than to seek additional facility routes, at least as a short-term recourse to meeting demand. There is thus a practical reason to be interested in transport network research that is especially focused on sparse transport graphs.

7.1 CHAINS AND LOOP-BACK

We can also easily observe that in bi-connected networks with $\bar{d} \ll 3$, there will be a preponderance of degree-2 nodes, and in the most extremely sparse networks, such nodes will tend to form chain sub-networks (or simply *chains*) like beads on a string, as is evident in many areas of the Level 3 network [75] as well as in those of [1], [43], [118]. If a number of degree-2 nodes in the network topology are all connected in series, then a chain is the logical collection of all such nodes and the spans incident upon them. By definition, chains are bounded at each end by nodes of degree $d \geq 3$, which we refer to as the *anchor nodes* of the chain. All spans within the chain are called *chain spans*. The extent to which ring-based networks have been deployed at the IXC level in North America compared to Europe is in a sense also a recognition of this sparseness in that rings are easily mapped onto these natural chains. However, rings have to be closed to operate, whereas a set of chain sub-networks could conceivably be operated at a higher level as a form of mesh-restorable network.

Simply increasing $\bar{d}$ by acquiring more rights of way is one way of improving the efficiency of low-connectivity mesh networks, but this is generally a very long-term and expensive option. In fact, rights-of-way costs are one of the single largest investments a network operator faces. Acquiring additional rights of way can easily involve years of legal work just to piece together the individual purchases and leases, municipal approvals, permits, etc., needed to establish a single new edge in the graph. Our specific aim here, therefore, is to find a way to enhance the efficiency of span-restorable mesh networks on low-degree topologies without modifying the underlying network topology.

As discussed above, one characteristic many sparse networks often exhibit is the concatenation of degree-2 nodes into chains. It is actually these chains that are the source of the poor capacity efficiency of the span restoration mechanism on sparse networks. Consider the failure of one of the spans on a chain. By its very nature, the span restoration mechanism will require a ring-like loop-back of any affected wavelengths through the chain in both directions. This will subsequently necessitate enough spare capacity on each span in the chain to meet or exceed the largest cross-section of working capacity anywhere within the chain.

This 100% matching of spare capacity to largest-working capacity is a general property of any degree-2 sub-network such as a ring or a chain of degree-2 nodes (the only structural difference between a ring and a chain is that a ring is a sub-network of degree-2 nodes arranged in a cycle, while a chain is a sub-network of degree-2 nodes that does not close upon itself). The main point to observe is that at any degree-2 site, the spare capacity on one side of the node must meet or exceed the working capacity on the other side of the same node, and vice-versa. The topology of a ring or chain dictates that to escape from a cut on one side of a node, the spare capacity on the other side must be sufficient to support loop-back of the failed working capacity on the cut side.

The notion of loop-back as an inescapable requirement in span restoration where chains are present and the 100% matching of working and spare capacity to support loop-back are the reasons behind the relative inefficiency of span restoration in a sparse network. We can demonstrate this by considering how restoration within chains is handled in a span-restorable network. Figure 7.2 shows a three-span chain and a set of working capacity accumulations (w_{Tot}) resulting from the routing of demands in the network. These w_{Tot} values may be either the resultant accumulation of demands crossing the span from shortest-path routing, or the corresponding totals from a joint capacity design, which does not necessarily route demands on shortest paths. Now consider the minimum spare capacity requirements of this chain. Under span restoration the entire chain must have spare capacity sufficient to support the loop-back re-routing of the span of the chain that has the largest working capacity cross-section. In other words, the conventional span restoration model will capacitate chains essentially as if they were sections of BLSR/OSPR-type rings. Thus, in the example, the worst-case cut is of the span between nodes 2 and 3 at 440 working units and so the spare capacity allocation on the other spans within the chain would be 440 as shown (note that span 2-3 only really needs 370 units of spare capacity since it must at worst accommodate restoration routes for the failure of span 1-2).

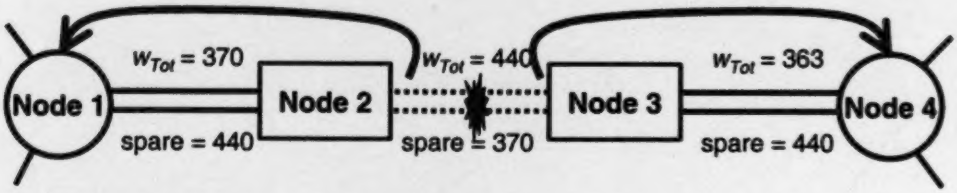


Figure 7.2 – The amount of spare capacity on a chain is determined by the size of the largest working capacity total on any span within the chain.

We can also note that despite the fact that the network as a whole relies on mesh OXCs to perform working and restoration routing in a span restoration environment, all nodes within chains could be equipped with OADMs instead since we are dealing exclusively with degree-2 nodes. This will result in no loss of flexibility since the only action they need to perform is a simple loop-back operation, and they would provide a cost savings compared to the more costly OXCs.

7.2 BREAKING DOWN WORKING CAPACITY INTO LOCAL AND EXPRESS COMPONENTS

A closer look at the makeup of the working traffic within a chain will show that it is a result of accumulations of working flows for some demands originating and/or terminating within the chain as well as others that pass completely through the chain on general paths across the network as a whole. We can consider the working capacity that arises from demands originating and/or terminating at one of the nodes within the chain (anchor nodes excluded) as *intra-chain* or *local flow* working capacity (w_{Loc}). The remaining working capacity on each chain span is therefore due to an accumulation of working flows arising from demands originating and/or terminating at the anchor nodes or nodes outside the chain. We call this *express flow* working capacity (w_{Exp}) since it is needed to accommodate express traffic whose composition isn't altered during its transit through the chain. An example of such a breakdown of working capacity is illustrated in Figure 7.3, where 211 wavelengths flow entirely through the chain on their way to/from other nodes located elsewhere in the network.

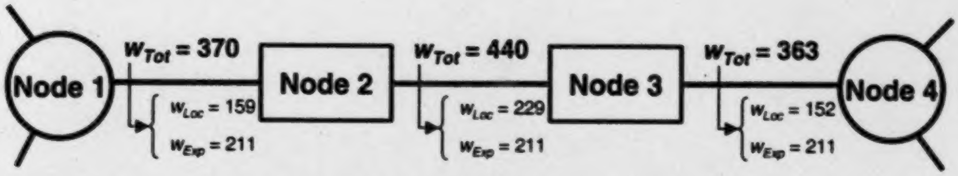


Figure 7.3 – Working capacity in a chain supports local and express flow of working routing.

Now we can observe that given the breakdown of working capacity into local and express flow through the chain, then conventional span restoration requires loop-back spare capacity for the entire cross-section of both local and express flows through the chain. The normal response to any span cut within the chain is therefore to return both local and express flows via loop-back to the anchor nodes. From there, the restoration re-routing problem can be viewed as equivalent to span restoration of a single edge failure in the remainder of the network graph. But because the treatment of restoration flows is completely mesh-like once they reach the anchor nodes, there is no need to explicitly loop back the express flows to the anchor nodes. Rather, the express component of the working flow crossing the failed span could be returned to the anchor nodes simply by letting those lightpaths fail all the way back to the anchor nodes. In other words, with respect to the express flows, the entire chain need only be viewed as one logical span over which these demands are travelling. In this view, failure of any real span within the chain is equivalent to failure of the corresponding logical span with respect to the express flows only. This is not the case for the local flow, however, because the set of demands affected by each span cut in the chain differs depending on which chain span is cut. The local demands must therefore be explicitly looped back to the anchor nodes because their aggregate composition is modified by add/drop actions within the chain itself. The composition of the express flows is, however, unchanged no matter where the cut occurs in the chain, so they can be simply failed all the way back to the anchor nodes and do not need loop-back.

The advantage in treating the express flows this way is that there will be no spare capacity required within the chain itself for restoration of any express flows, other than an allocation of spare that may be made to the chain from a globally efficient standpoint. We demonstrate in Figure 7.4 that this opportunity to treat express flows as failures requiring spare capacity external to the chain only, leaving the chain spare capacity to be driven exclusively by the considerations of local flow loop-back, can provide quite substantial reductions in spare capacity requirements. Here, local working flows must still be matched by loop-back spare capacity so they can escape back to the anchor nodes with the demand composition they had at the particular location of the break. On the other hand, since there is no change in their composition within the chain, all express flow through the chain can automatically fail back to the anchor nodes and be treated entirely with mesh-based restoration principles in the remainder of the network; it never enters into the spare capacity sizing of the spans within the chain. Now, since the 211 units of express flow working capacity no longer requires loop-back spare capacity within the chain, then the spare capacity only needs to accommodate the maximum of the local flows, which is 229 units (rather than the original 440 units). We note that the spare capacity requirements in the network external to the chain remain unchanged since both the local and express flows still need to be rerouted between the anchor nodes whether they were both looped back within the chain or not.

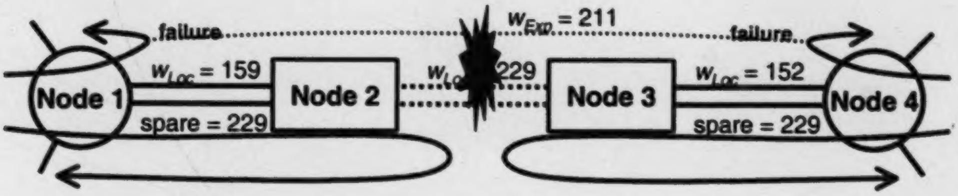


Figure 7.4 – Allowing express flows to fail all the way back to the anchor nodes reduces loop-back spare capacity required within the chain.

7.3 THE META-MESH

Having made those observations of the local conditions inside a chain, we now move out to view the chain as a constituent part of a *meta-mesh* network. The meta-mesh is not a different layer network, nor is it a sub-network, but rather it is simply the graph topology that is formed when we view each chain sub-network as equivalent to a single span between its anchor nodes. In other words, the meta-mesh is the topology obtained when nodes of only degree-3 or higher are considered and each degree-2 node is collapsed so that the two spans incident on it are combined into a single span. In the meta-mesh, we make no distinction between a span connecting degree-3 or higher nodes and chain sub-networks, they are both just logical spans of the meta-mesh. The meta-mesh therefore corresponds to a homeomorphism of the original network topology (refer back to Section 2.2). The meta-mesh topology of the Level 3 network from Figure 7.1 is illustrated in Figure 7.5.

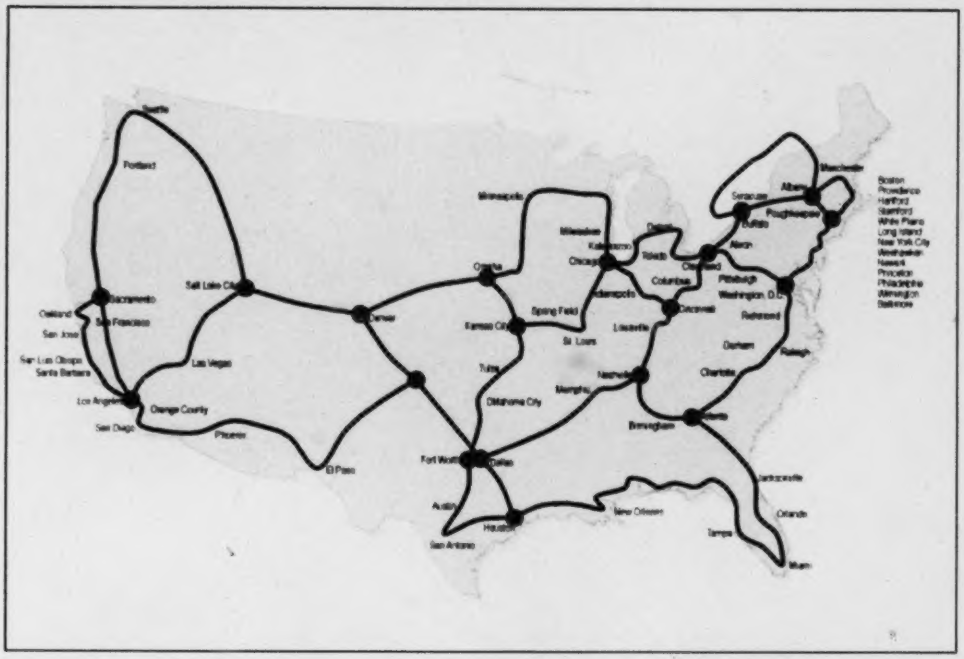

Seattle
Portland
Sacramento
Oakland
San Francisco
San Jose
San Luis Obispo
Santa Barbara
Los Angeles
Orange County
San Diego
Phoenix
Las Vegas
Salt Lake City
Denver
El Paso
Minneapolis
Omaha
Kansas City
Tulsa
Oklahoma City
Fort Worth
Dallas
Austin
San Antonio
Houston
Milwaukee
Kalamazoo
Chicago
Spring Field
St. Louis
Indianapolis
Detroit
Toledo
Columbus
Cleveland
Cincinnati
Louisville
Nashville
Memphis
Birmingham
Atlanta
Akron
Pittsburgh
Washington D.C.
Richmond
Durham
Raleigh
Charlotte
Jacksonville
Orlando
Tampa
Miami
New Orleans
Syracuse
Buffalo
Albany
Manchester
Poughkeepsie
Boston
Providence
Hartford
Stamford
White Plains
Long Island
New York City
Weehawken
Newark
Princeton
Philadelphia
Wilmington
Baltimore

Figure 7.5 – The meta-mesh of the Level 3 Communications, Inc. USA backbone network.

The significance of the meta-mesh is that it is only at this level of abstraction that true span-restorable spare capacity sharing efficiencies *can* arise. While the complete network has 71 nodes, 83 spans and $\bar{d}$ = 2.34, the meta-mesh graph example has only 19 nodes and 31 spans with $\bar{d}$ = 3.26. We note that by its very nature, the meta-mesh graph is always $\bar{d} \geq 3$ since the only spans remaining are of degree-3 or higher. The implications of this are seen when we apply the $1/(\bar{d}-1)$ lower bound on redundancy in a span-restorable network. In the original network, redundancy is limited to no less than $1/(2.34-1) = 75\%$. On the other hand, the redundancy of a span-restorable network designed on the meta-mesh version of the network could potentially be as low as $1/(3.26-1) = 44\%$. While these numbers represent lower bounds only and are not achievable in general, they demonstrate the significant potential for efficiency increases if we could somehow design a span-restorable network within the meta-mesh topology, rather than within the original sparsely connected network topology.

7.3.1 AUGMENTING THE TOPOLOGY WITH LOGICAL CHAIN BYPASSES

We now revisit the earlier discussion about treating express flow within chains separately from local flow as described in Section 7.2. To implement that scenario, the express flow must be allowed to simply fail end-to-end within the chain and be detected as having failed by the OXCs at the anchor nodes, rather than by OXCs or OADMs on either end of the actual failed span inside the chain. The presence of active loop-back signals in the spare ports at the anchor nodes can identify these ports as the local targets for restoration path substations. Both types of failed working flow, once at the anchor nodes through either direct fail-back (express flow) or loop-back (local flow), are then logically unified as a single total amount of failed working capacity on the corresponding span of the meta-mesh. From that point on, the anchor nodes will co-operate as a conventional pair of Sender-Chooser nodes (refer to the SHN protocol described in [45] and [47]), within the meta-mesh of OXC nodes. When restoration paths are found through the meta-mesh of which this chain is a part, the anchor nodes of the chain perform the standard span restoration function of making cross-connections that substitute the restoration paths between the corresponding ports at their locations. Thus, one logical mesh-restoration event between the chain's anchor nodes in the meta-mesh will transparently look after the simultaneous restoration of both the local and express flows within the chain.

Alternatively, we can think of the express flow as being routed over a single *logical chain bypass span*, whose length/cost is equivalent to the sum of the lengths/costs of all of the spans within the chain. The express flow is still routed over the chain, but is not terminated or even accessed at any point between the anchor nodes. It still follows the same physical route of the chain and nominally uses the same fibres, cables, etc., but is not implicitly handled by the OADMs en route. It is accessed only by the OXCs at the anchor nodes. Express flows could even be routed through straight fibre splices or glass-throughs at each node within the chain, and physically bypass the OADMs altogether. Ultimately, given sufficient express flow to warrant it, separate fibres could be employed to act as the chain bypasses so the anchor nodes need not differentiate between local and express flows. When a span within the chain fails, the anchor nodes would see a bypass span failure and act as a Sender-Chooser pair for the express flows routed over it, while the end-nodes of the failed span would see the actual span failure and act as a Sender-Chooser pair for the local flow routed over it. The anchor nodes would simultaneously act as tandem nodes for the restoration of the local flow and both flows would in effect be logically unified for restoration between the OXCs at the anchor nodes.

Now, from the point of view of express flows passing through any of the chains in the network, they are effectively routed within the meta-mesh, which operates at the reduced lower bound on redundancy that applies to the more richly connected topology. The local flows, on the other hand, see no change in the way they are handled or restored.

7.4 META-MESH DESIGN MODEL

We can apply the meta-mesh concept in a new network design model by making several changes to the conventional span-restorable JCA design model. First, the network topology must be augmented to include a logical bypass span in parallel with each chain sub-network. If a chain composition is (by nodes) A-B-C-D-E-F, with a total cost or length X, then the associated bypass span added to the topology is a new span with end-nodes A-F and an equivalent cost or length of X. Actually adding the bypass spans to the topology represents the possibility of routing working flows over an express route through the chain. If a demand originates or terminates at one of the nodes within the chain, the model will be forced to route it into the chain itself and would preclude the use of the bypass span. This implies its participation in the loop-back spare capacity required on the spans within the chain. However, if a demand neither originates nor terminates inside the chain, but is nevertheless required to pass over the chain in order to maximize global efficiency, then the bypass span represents an equidistant routing option that does *not* have the side effect of contributing to the loop-back spare capacity. The revised formulation will not explicitly *require* the solver to use the bypass spans, but rather under global minimization of total capacity, the solver will be further enabled to reduce total cost by the option to treat express flows in this separate way and eliminate the extra spare capacity that would otherwise be needed.

A side effect of routing express flows on the bypass spans is an implicit *grooming* benefit. Grooming is the long established technique of selecting and grouping demands that share a common destination (or next-hub en-route) onto the same carriers to reduce the nodal equipment needed. In this sense the implicit action of the solver in the presence of the bypass spans is a special instance of grooming in WDM networks [84], which further reduces equipment counts. Here, the nodal equipment reductions arise because express demands do not consume interfaces or core bandwidth in the OADMs along the chain. The grooming effect is separate from the benefit of spare capacity reduction through the loop-back argument but is automatically captured by the aspect of jointness in the formulation.

The span-restorable JCA arc-path design model in Section 5.1.1 is then extended to convert single physical cuts of spans within chains into corresponding logical dual failure scenarios where there is simultaneous failure of the associated bypass span between the anchor nodes of that chain. To represent these simultaneous logical span failures, we first introduce new notation as follows:

New Sets:

- $S_D \subseteq S$ is the set of all *direct* spans in the network, or spans whose end-nodes are both of degree-3 or higher.
- $S_B \subseteq S$ is the set of all chain bypass spans added to the network.
- $S_C \subseteq S$ is the set of all spans that are a part of any chain in the network, and is equivalent to $S-(S_D \cup S_B)$.

New Input Parameters:

- $k_i \in S_B$ is the many-to-one mapping between individual spans of the full network and an associated logical bypass *k*. For instance, if spans S7, S8, S9, S11, and S12 comprise the chain whose bypass span is B6, then $k_{S7} = k_{S8} = k_{S9} = k_{S11} = k_{S12} = B6$.

In addition to the new notation, all previous notation from the original span-restorable JCA design model formulation in Section 5.1.1 remains. The formulation itself is expressed as follows.

$$\text{Minimize} \quad \sum_{\forall j \in S} c_j \cdot \left(s_j + w_j\right) \tag{7.1}$$

$$\text{Subject to:} \quad \sum_{\forall q \in Q^r} g^{r,q} = d^r \qquad \forall r \in D \tag{7.2}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r} \zeta_j^{r,q} \cdot g^{r,q} = w_j \qquad \forall j \in S \tag{7.3}$$

$$\sum_{\forall p \in P_i} f_{i,p} = w_i \qquad \forall i \in S \tag{7.4}$$

$$s_j \geq \sum_{\forall p \in P} \delta_{i,j}^p \cdot f_i^p \qquad \forall i \in S_d \quad \forall j \in S \middle| i \neq j \tag{7.5}$$

$$s_j \geq \sum_{\forall p \in P_i} \delta_{i,j}^p \cdot f_i^p + \sum_{\forall p \in P_{k_i}} \delta_{k_i,j}^p \cdot f_{k_i}^p \qquad \forall i \in S_c \quad \forall j \in S \middle| i \neq j \neq k_i \tag{7.6}$$

The objective function in equation (7.1) is the same as that for the original JCA formulation and seeks to minimize the cost of working and spare capacity in the network design. Constraint sets (7.2), (7.3), and (7.4) are identical to constraint sets (5.5), (5.6), and (5.2), respectively, from the original span-restorable JCA formulation, and ensure the proper working routing, working capacity placement, and restoration routing, respectively. However, the prior sets of eligible working routes, Q^r, and eligible restoration routes, P_i, are now regenerated within the augmented topology that includes the chain bypass spans. Q^r is initially enumerated within the original topology only and then for each eligible working route that fully transits a chain, Q^r is supplemented with the equivalent-length route that crosses that chain's bypass span. In cases where an eligible working route fully transits multiple chains, all combinations of routes crossing those chains' bypasses are included. P_i is structured to recognize and accommodate the logical dual-failure combinations that now arise when a chain span fails. For all spans in the original topology that are not a part of a chain, no special considerations are needed, so for these spans, enumeration of P_i

effectively remains unchanged from the equivalent P_i in the conventional JCA design model. However, eligible route sets for spans within a chain are restricted so that P_i does not include routes over the associated (and co-failed) bypass span. Likewise, eligible route sets for bypass spans are enumerated so that they do not include spans in the associated chain.

The original constraint set in equation (5.3) from the conventional span-restorable JCA network design formulation is modified here to capture the logical dual-failure scenarios that arise when a chain span is cut (the chain's associated bypass span is also considered to have simultaneously failed). That constraint set is split into the two associated constraint sets shown in equations (7.5) and (7.6) so that spans within chains can be handled separately (and differently) from the direct (non-chain) spans. The constraint set in equation (7.5) ensures that there is sufficient spare capacity on any span *j* to accommodate all restoration flow routed over it for failure of any non-chain span in the original network topology. This equation is identical to the constraint set in equation (5.3), with the exception that $\forall i \in S_d$ now applies rather than $\forall i \in S$. Equation (7.6) ensures that there is sufficient spare capacity on any span *j* to support all the restoration flows routed over it for the simultaneous failure of any chain span *i* as well as its associated bypass span k_i.

7.5 EXPERIMENTAL STUDY METHOD

To validate these ideas and to characterize the capacity and equipment savings relative to the conventional span-restorable JCA design formulation (and the path-oriented schemes), we conducted comparative capacity design trials using the test-case network topologies described in Section 6.1. As with the benchmarking designs in CHAPTER 6, the meta-mesh design model was implemented in AMPL Mathematical Programming Language version 9.0 and solved with the CPLEX 9.0 MIP Solver on a 4-processor SUN UltraSparc III running at 900 MHz with 16 GB of RAM. Pre-processing for eligible working and restoration routes, and other input parameters was done on a dual-processor AMD Opteron 242 PC with 1 GB of RAM, running Windows 2000. All working and spare capacity allocations were integer, corresponding to capacity design and restoration mechanisms at the wavelength level. Results are based on a full CPLEX termination with a MIPGAP setting of 10^{-3} (i.e., solutions are guaranteed to be within 0.1% of optimal). All took less than two minutes to solve, and, except for the most richly connected networks from the 40n80s1 network family, they actually took no more than several seconds.

Pre-processing of the eligible working route sets initially enumerated the 5 shortest routes between each O-D node pair within the original topology only (no bypass spans considered). For each eligible working route that fully transits a chain, the eligible working route set was then supplemented with the equivalent route that crosses that chain's bypass span. In cases where an eligible working route fully transits multiple chains, all combinations of routes crossing those chains' bypasses are included. Pre-processing of eligible restoration route sets enumerated the 10 shortest routes between the end-nodes of each span within the original topology only. Since we provided no explicit benefit in the model for a restoration route to use a chain's bypass rather than the chain itself, adding eligible restoration routes that use bypass spans will have no effect other than to increase the complexity of the ILP (more spans and more eligible restoration routes results in more constraints and variables in the ILP).

7.6 RESULTS AND DISCUSSION

Results of meta-mesh design experiments are presented and discussed in the coming sections. We first examine meta-mesh network capacity requirements in the same manner as the various benchmark survivability mechanisms were studied in Section 6.4, and then compare it to span-restorable JCA designs, which are the basic mechanism most similar to meta-mesh.

7.6.1 META-MESH CAPACITY COSTS

7.6.1.1 TOTAL CAPACITY COSTS

Figure 7.7 through Figure 7.12 show normalized total capacity costs of optimally designed (i.e., minimum cost) networks of the various families designed to be 100% restorable using meta-mesh restoration, as well as the equivalent designs using the JCA models for span restoration, *p*-cycle restoration, SBPP, and path restoration. Each figure provides data for a single network family, and each data point represents the total wavelength-kilometres of working and spare capacity required to route all demands and provide for full restoration in the optimal solution for the member of the family with the indicated average nodal degree, $\bar{d}$, using the specified survivability mechanism. In each figure, capacity costs have been normalized to that of the lowest-cost design over all networks in the family and all survivability mechanisms. Within each figure, data points have been organized into curves corresponding to the various survivability mechanisms.

We first note that the meta-mesh curves do not cover the entire range of $\bar{d}$ for any of the network families, due to the manner in which meta-mesh restoration is performed. Meta-mesh restoration differs from conventional span restoration in that working routes affected by failure of a span *on a chain* are allowed to fail back to the anchor nodes of the chain, but in the more densely connected networks, no such chains exist, and so meta-mesh restoration is equivalent to span restoration. There are also two sparse test cases (25n50s1-25s and 25n50s1-26s) where meta-mesh is identical to span restoration. In 25n50s1-25s, the spans in the network comprise a Hamiltonian cycle, and there is no chain per se, or rather, the entire network is a chain and there are no anchor nodes to which a working route can fail. So in this trivial case, any span failure will require loop-back capacity between the end nodes of the failure and the end-nodes of the working route. The 25n50s1-26s network topology differs from 25n50s1-25 by the addition of a single span, which partitions the Hamiltonian cycle into two true chains, as shown in Figure 7.6. This too, however, is a trivial case from the point of view of meta-mesh restoration since that single added span acts as a naturally occurring bypass for the two chains. Conventional span restoration will therefore spontaneously make use of that span in exactly the same manner as meta-mesh restoration would if we'd added our own artificial bypass spans.

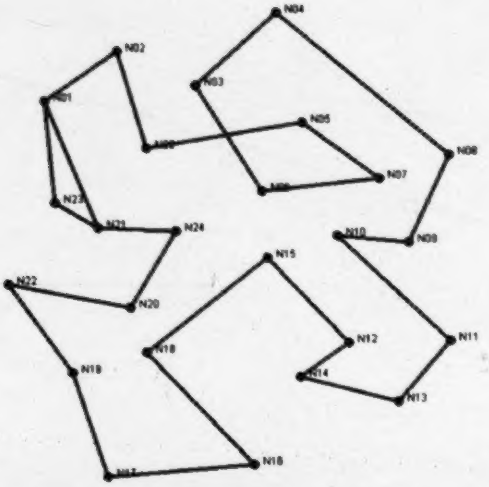


Figure 7.6 – Test case 25n50s1-26s network topology.

We can observe qualitatively from Figure 7.7 through Figure 7.12 that the meta-mesh network designs require more capacity than the path restoration and SBPP designs and less than the span restoration and *p*-cycle designs. On average over all 163 test case networks (124 actually, since meta-mesh is equivalent to span restoration in the 39 networks with no degree-2 nodes and in the two special cases mentioned above), meta-mesh restoration requires 1.9% less total capacity than span restoration, 8.8% more total capacity than SBPP, and 12.6% more total capacity than path restoration. In two network families (20n40s1 and 30n60s1), meta-mesh restoration provides only 0.8% total capacity costs savings relative to span restoration. In the others, however, the meta-mesh designs perform substantially better. In fact, in the 15n30s1 network family, optimal meta-mesh-restorable designs require an average of 4.8% less total capacity than the equivalent span-restorable designs. In some test case network topologies, total capacity requirements in meta-mesh designs were as much as 8.4% less costly than those of span restoration, and in five test cases, meta-mesh outperformed SBPP, and came as close as 0.6% from path restoration (which is within the path restoration design gap to optimality). In general, meta-mesh tended to perform better relative to the other survivability mechanisms when the network topology was sparse. This will be discussed more fully in subsequent sections of this chapter.

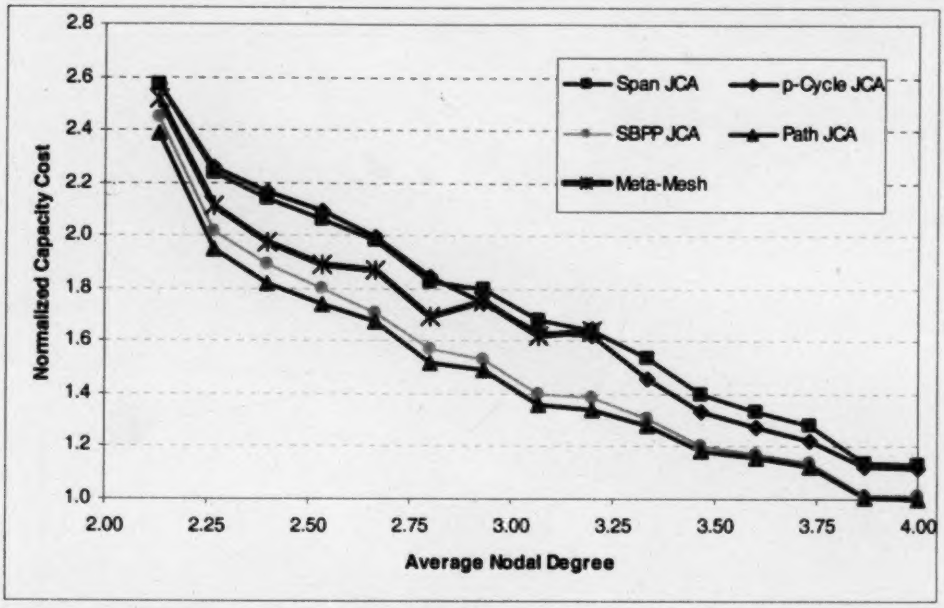


Figure 7.7 – Meta-Mesh normalized total capacity costs for the 15n30s1 network family.

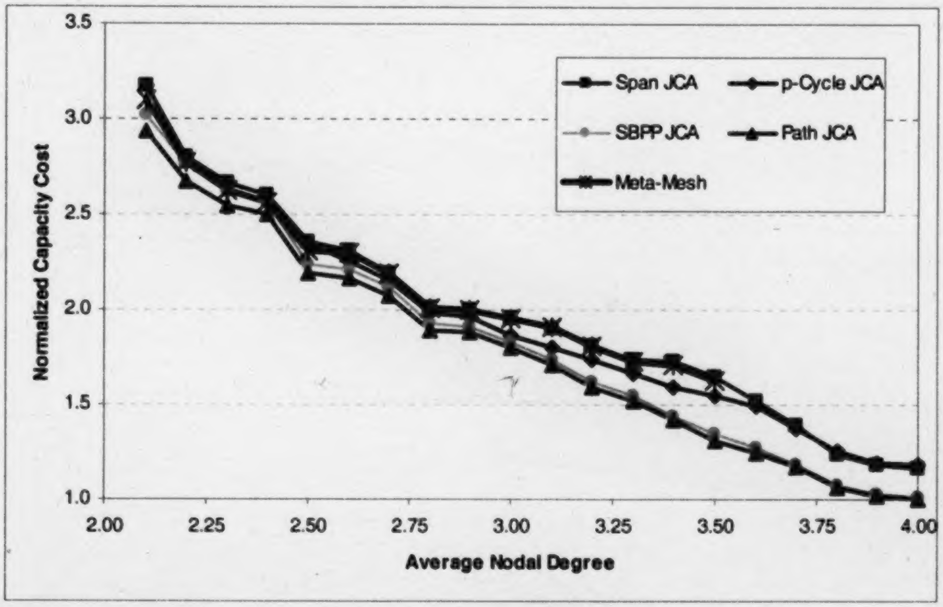


Figure 7.8 – Meta-Mesh normalized total capacity costs for the 20n40s1 network family.

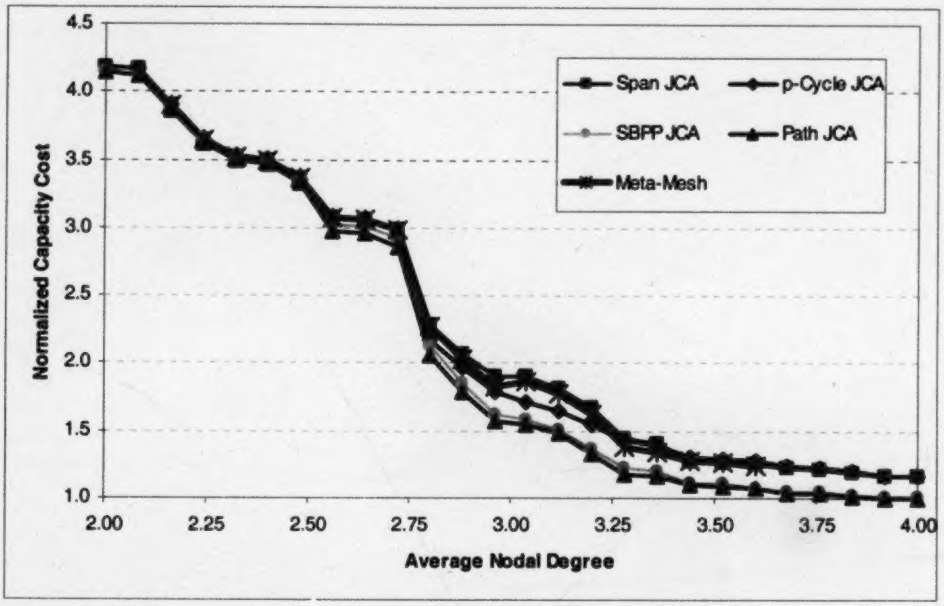


Figure 7.9 – Meta-Mesh normalized total capacity costs for the 25n50s1 network family.

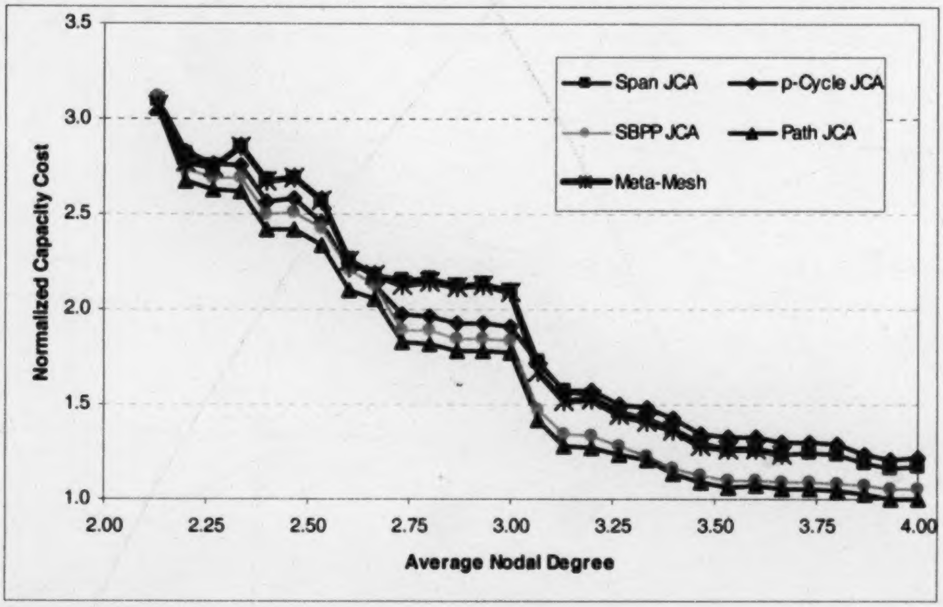


Figure 7.10 – Meta-Mesh normalized total capacity costs for the 30n60s1 network family.

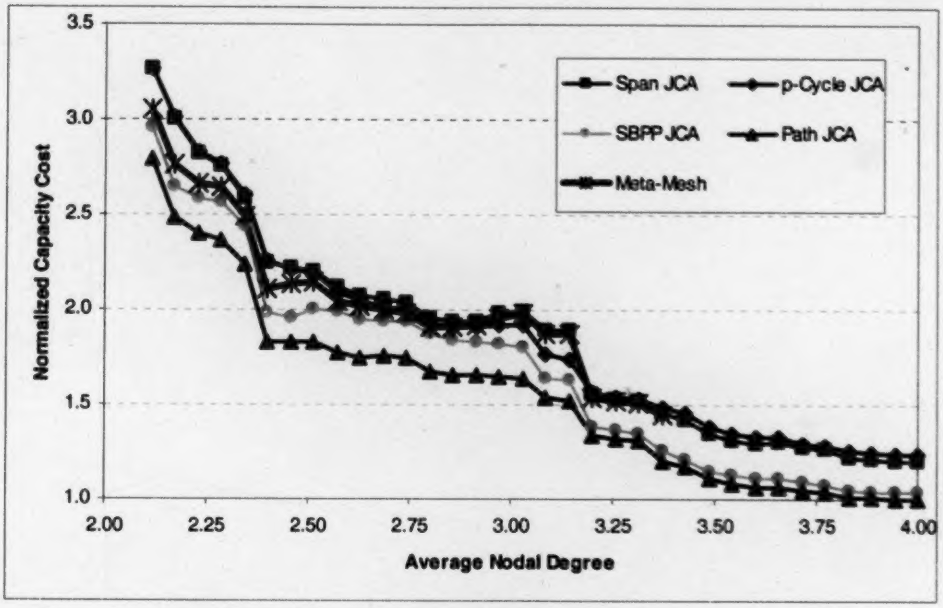


Figure 7.11 – Meta-Mesh normalized total capacity costs for the 35n70s1 network family.

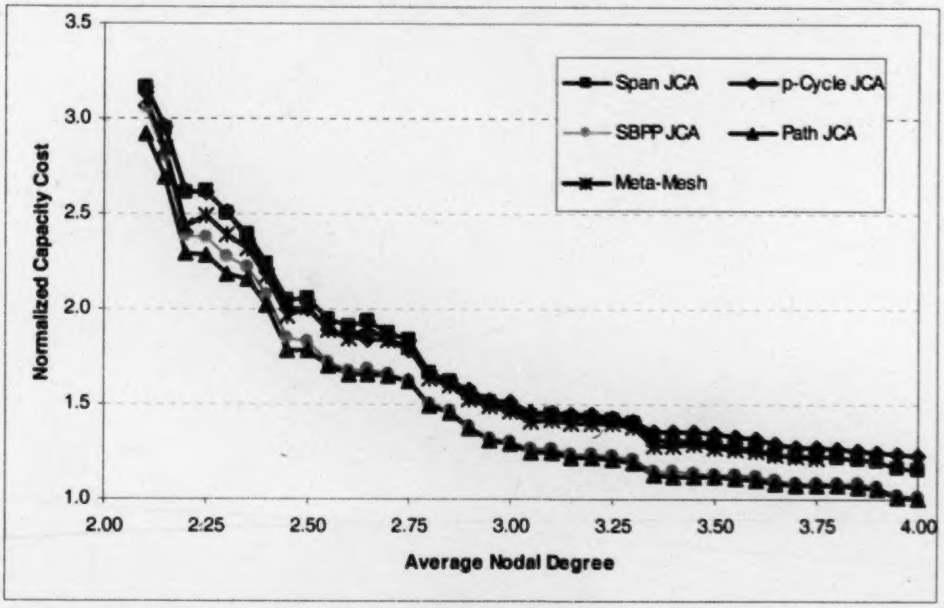


Figure 7.12 – Meta-Mesh normalized total capacity costs for the 40n80s1 network family.

7.6.1.2 SPARE CAPACITY COSTS

Figure 7.13 through Figure 7.18 show normalized spare capacity costs of optimally designed networks of the various families designed to be 100% restorable using meta-mesh restoration, as well as the equivalent designs using the JCA models for span restoration, *p*-cycle restoration, SBPP, and path restoration. Each figure provides data for a single network family, and each data point represents the total wavelength-kilometres of spare capacity required to provide for full restoration in the optimal solution for the member of the family with the indicated average nodal degrees, $\bar{d}$, using the specified survivability mechanism. In each figure, spare capacity costs have been normalized to that of the lowest-cost design over all networks in the family and all survivability mechanisms. Within each figure, data points have been organized into curves corresponding to the various survivability mechanisms.

In Section 6.4, we noticed that differences between various survivability mechanisms were most evident when comparing their spare capacity costs. The same is true of meta-mesh. When considering spare capacity costs only, meta-mesh is 3.9% less costly than span restoration, and in 10 of the test networks, meta-mesh required in excess of 10% less spare capacity than span restoration. In the 15n30s1 network family, meta-mesh needs an average of 9.6% less spare capacity than span restoration, and in the 35n70s1 family, it needs 6.1% less on average. And although the meta-mesh designs use an average of 22.9% more spare capacity than SBPP, there is one network topology (30n60s1-32s) where meta-mesh requires 5.1% *less* than SBPP, one (35n70s1-49s) where meta-mesh uses 2.6% less spare capacity, and another (35n70s1-45s) where it needs 1.1% less.

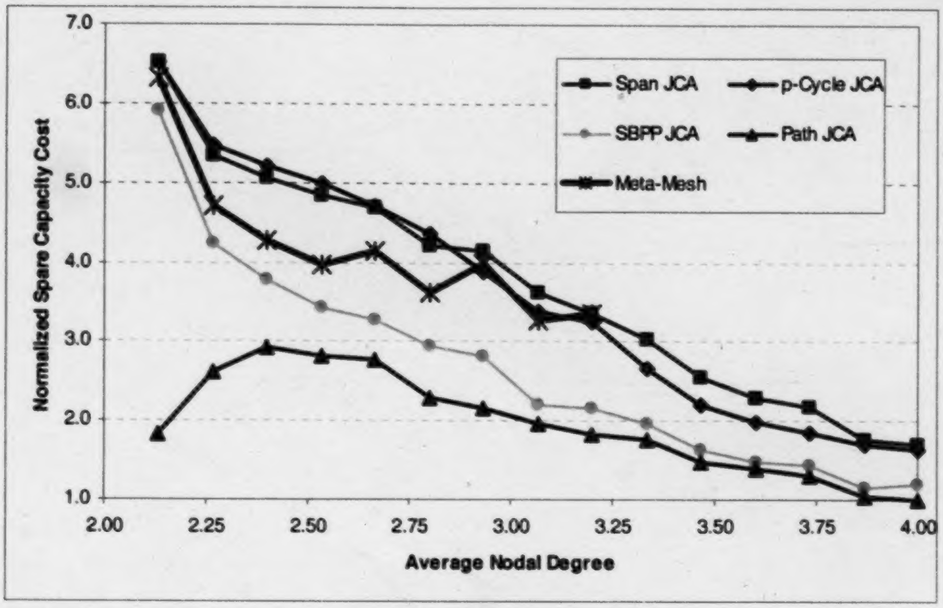


Figure 7.13 – Meta-Mesh normalized spare capacity costs for the 15n30s1 network family.

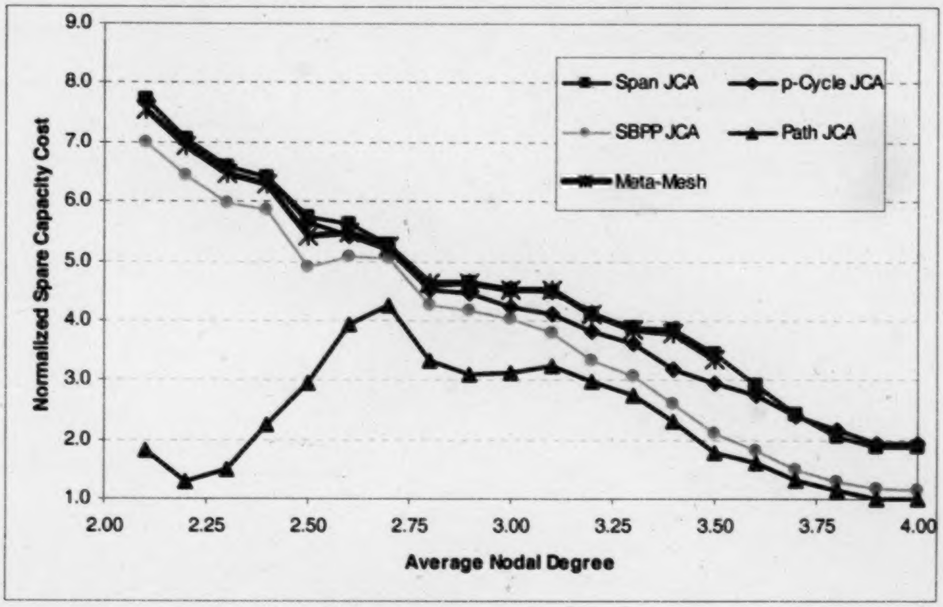


Figure 7.14 – Meta-Mesh normalized spare capacity costs for the 20n40s1 network family.

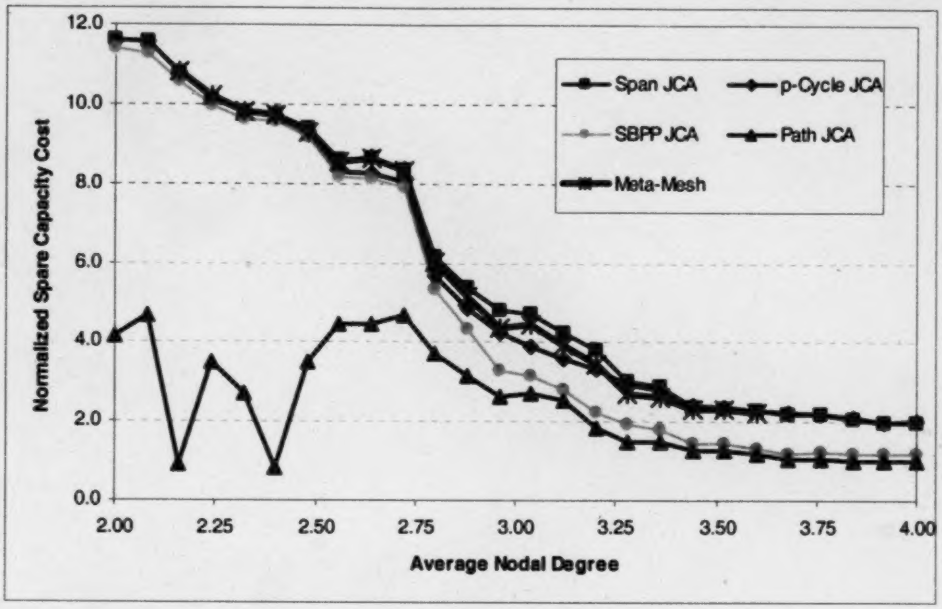


Figure 7.15 – Meta-Mesh normalized spare capacity costs for the 25n50s1 network family.

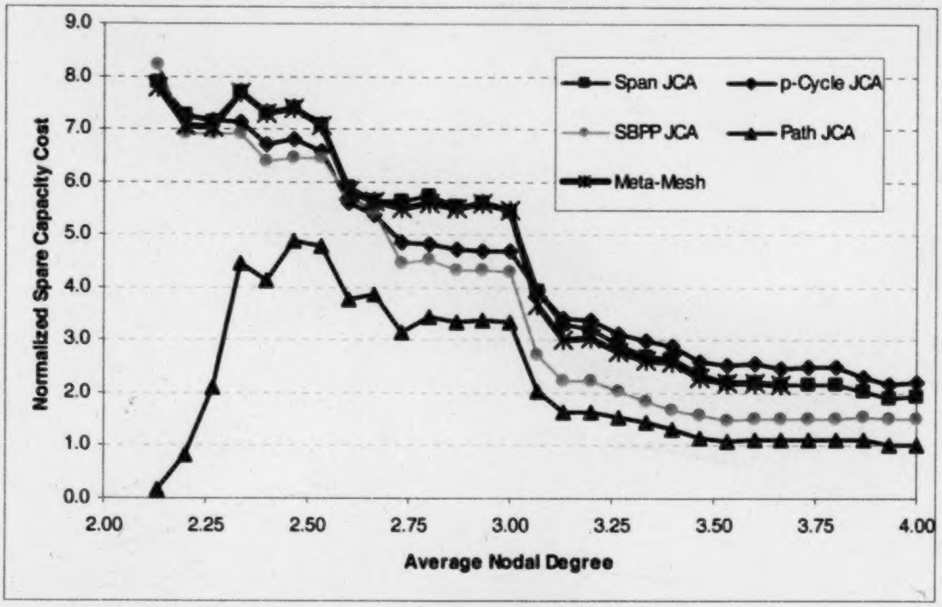


Figure 7.16 – Meta-Mesh normalized spare capacity costs for the 30n60s1 network family.

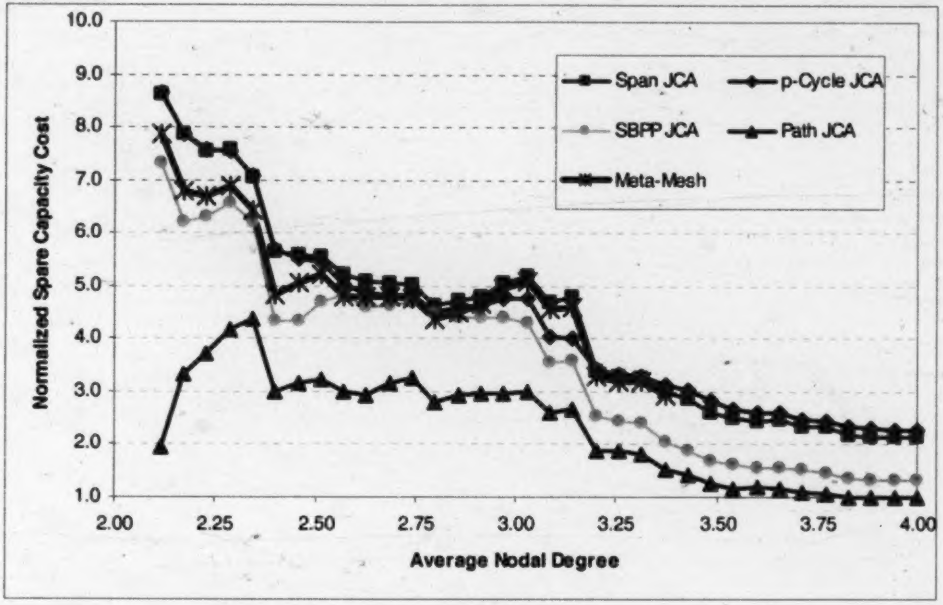


Figure 7.17 – Meta-Mesh normalized spare capacity costs for the 35n70s1 network family.

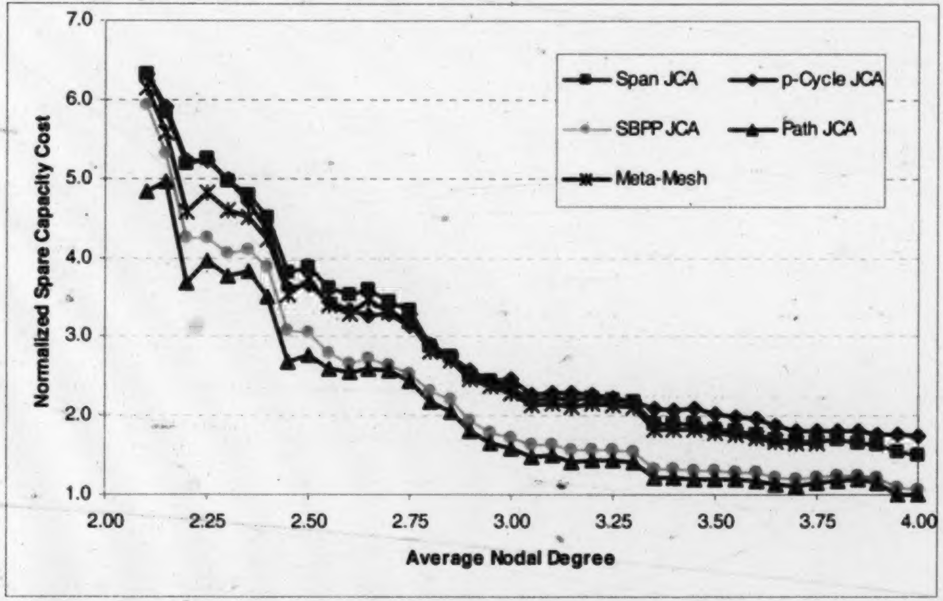


Figure 7.18 – Meta-Mesh normalized spare capacity costs for the 40n80s1 network family.

7.6.1.3 CAPACITY COST REDUNDANCIES

Figure 7.19 through Figure 7.24 show capacity cost redundancies of optimally designed networks of the various families designed to be 100% restorable using meta-mesh restoration, as well as the equivalent designs using the JCA models for span restoration, *p*-cycle restoration, SBPP, and path restoration. Like the capacity cost figures already seen, each figure provides data for a single network family, and each data point represents the R_{cost} of the optimal solution for the member of the family with the indicated average nodal degree using the specified survivability mechanism. The $1/(\bar{d}-1)$ lower bound on capacity redundancy in a span-restorable network has been added to each figure for reference and comparison purposes.

On average, the capacity cost redundancies of meta-mesh designs are 3.8% below those of span-restorable designs, and in 14 of the test cases, meta-mesh redundancies are in excess of 10% below span restoration. In one test case (15n30s1-19s), the capacity cost redundancy of the meta-mesh design was 18.1% below the redundancy of the span-restorable design. Meta-mesh redundancies were an average of 21.8% above those of SBPP, but there are six test cases where meta-mesh redundancies are even *below* SBPP (in 35n70s1-49s it was 7.9% below).

Perhaps the most interesting observation we can make is that in several test cases, meta-mesh redundancy very closely approaches the $1/(\bar{d}-1)$ lower bound on capacity redundancy in a span-restorable network. While it doesn't quite break below it in any of the test cases studied here, there are several cases in a 32-node network family studied in [25] and [54] where meta-mesh redundancy actually does breach $1/(\bar{d}-1)$. So how is this explained? The answer is that a meta-mesh design actually represents a partial step toward path restoration. Local flows originating or terminating within a chain struck by a span failure are restored in exactly the same way in both span restoration and meta-mesh restoration. But the express flows transiting the entire chain are restored between the chain's anchor nodes, which serve as a kind of pseudo-origin-destination node pair part of the way towards the affected lightpath's actual end-nodes. Moreover, for express flows between the anchor nodes themselves (i.e., the lightpath's end-nodes are the chain's anchor nodes), the resultant meta-mesh restoration is actually identical to path restoration for the stipulated demands. Therefore, to the extent that chains are a significant consideration in sparse networks, the meta-mesh method effectively moves the network design toward the efficiency of a path-restorable design on the same topology. Thus, span restoration on the meta-mesh abstraction of a sparse graph can approach the behaviour of path restoration on the full graph, even though it continues to use or require only a span restoration mechanism.

Alternatively, we can refer back to the initial argument about restoring express flows within the highly connected meta-mesh equivalent topology (i.e., where chains are replaced by logical bypass spans) rather than in the original sparse topology. From the point of view of those express wavelengths, the lower bound on their redundancy can be calculated based on the average nodal degree of the meta-mesh graph. So if a network with $\bar{d} = 2.5$ has a meta-mesh equivalent with $\bar{d} = 3.5$, then those express wavelengths are effectively bounded by a minimum redundancy of $1/(\bar{d}-1) = 40\%$, rather than the $1/(\bar{d}-1) = 66.7\%$ minimum redundancy of local flow.

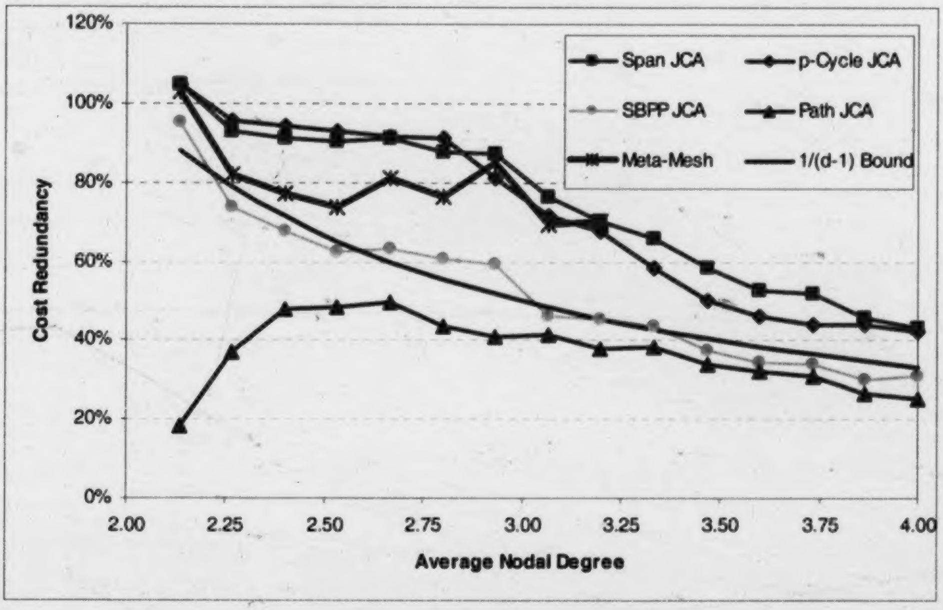


Figure 7.19 – Meta-mesh capacity cost redundancy for the 15n30s1 network family.

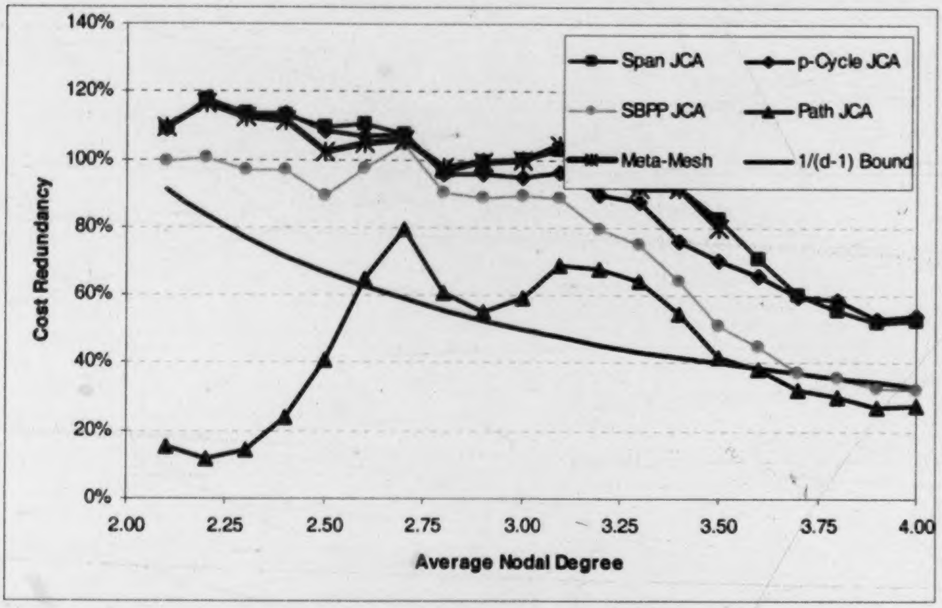


Figure 7.20 – Meta-mesh capacity cost redundancy for the 20n40s1 network family.

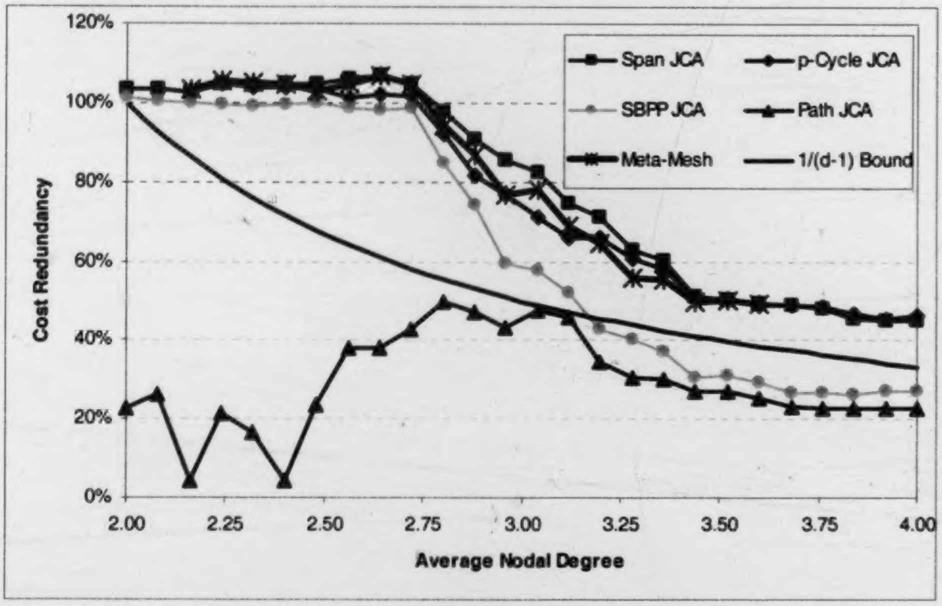


Figure 7.21 – Meta-mesh capacity cost redundancy for the 25n50s1 network family.

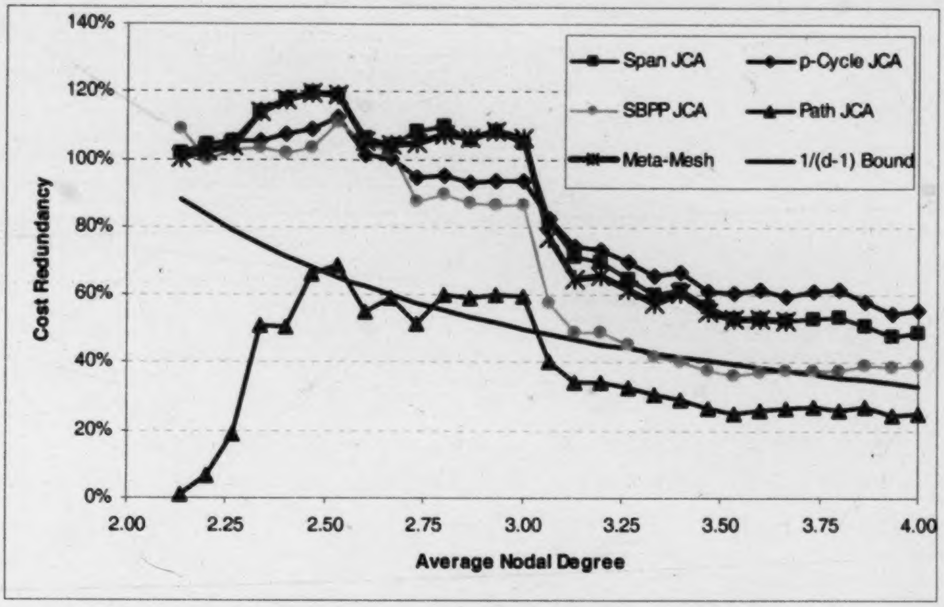


Figure 7.22 – Meta-mesh capacity cost redundancy for the 30n60s1 network family.

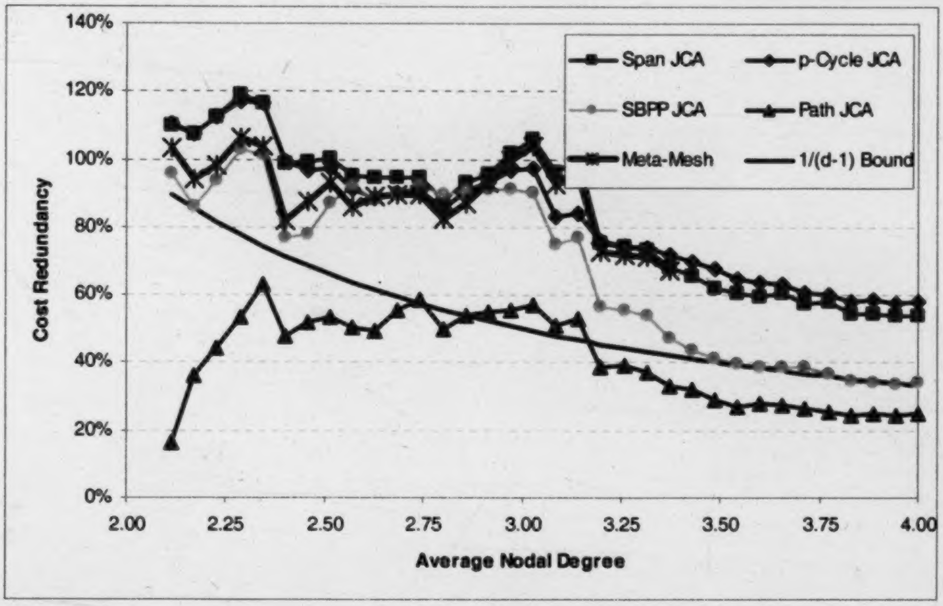


Figure 7.23 – Meta-mesh capacity cost redundancy for the 35n70s1 network family.

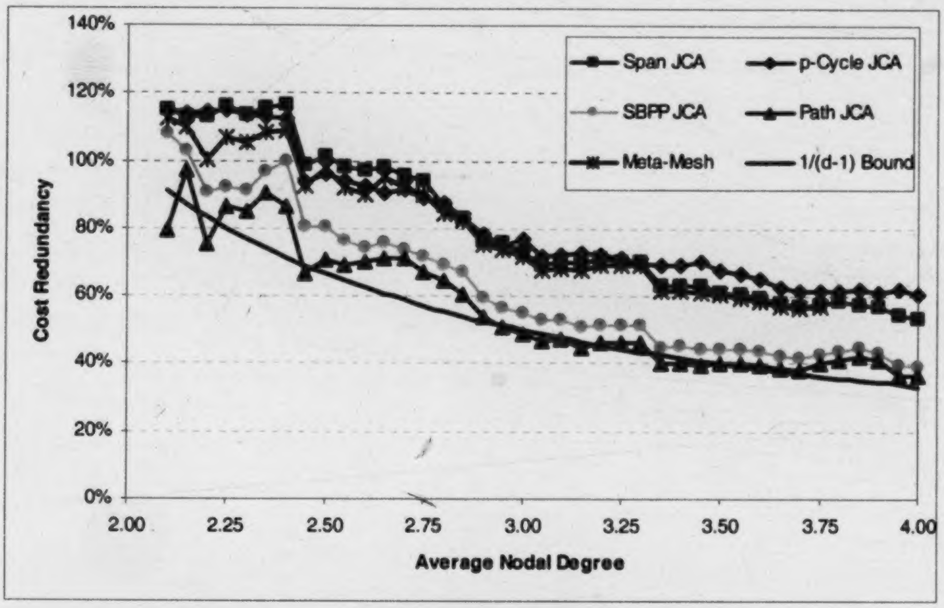


Figure 7.24 – Meta-mesh capacity cost redundancy for the 40n80s1 network family.

7.6.2 META-MESH VERSUS SPAN-RESTORABLE JCA

Since meta-mesh is essentially an improvement on span restoration that allows for a more efficient use of capacity in a sparse network, we now take a closer look at how meta-mesh compares to span restoration in terms of capacity costs and redundancy.

7.6.2.1 CAPACITY COST REDUCTION

Figure 7.25 through Figure 7.30 show the capacity cost reduction relative to span restoration and the percentage express flow in optimally designed meta-mesh-restorable networks. Each figure provides data for a single network family. Each data point in the blue curve represents the percentage difference between the total wavelength-kilometres of working and spare capacity required to route all demands and provide for full restoration in the optimal meta-mesh design solution relative to the equivalent span-restorable JCA design for the member of the family with the indicated average nodal degree, $\bar{d}$. Each data point in the green curve represents the percentage difference between the total wavelength-kilometres of spare capacity required to provide for full restoration in the optimal meta-mesh design solution relative to the equivalent span-restorable JCA design. Data points in the red curve correspond to the percentage of all working capacity (on a wavelength-kilometre basis) in a meta-mesh design that is express flow fully transiting a chain.

Experiments show that meta-mesh improved upon the span-restorable JCA designs by an average of 3.9% reduction in spare capacity and 1.9% reduction in total capacity. As discussed in Section 7.6.1, there were 10 test cases where the spare capacity reductions exceeded 10%, and for one network (15n30s1-19s) meta-mesh required 18.1% less spare capacity than span restoration. There were also 10 test case networks where meta-mesh total capacity requirements were at least 5% less than span restoration. Some network families experienced greater capacity reductions than others. For instance, spare capacity and total capacity of the 15n30s1 network family were reduced an average of 9.6% and 4.8%, respectively, and the 35n70s1 network family saw an average of 6.1% savings in spare capacity and 2.7% savings in total capacity. There are also a few test cases (eight to be exact), where spare capacity costs were slightly greater in the meta-mesh designs than they were in the corresponding span-restorable designs. In most cases, the difference was within the design solution gap to optimality, and in all cases, slight decreases in working capacity costs more than compensated for the spare capacity cost increases, which was only 0.78% in the worst case.

We can also note that meta-mesh *working* capacities (not shown) are an average of only 0.05% greater than those of span restoration JCA designs, and ranged from 2.8% below and 3.1% above the span restoration JCA working capacities. Even the absolute difference between meta-mesh and span restoration JCA working capacities averaged only 0.65%. In most cases, working routing differed very little between meta-mesh and span restoration JCA designs (no more so than any two successive span-restorable JCA designs of the same test network) suggesting that any capacity savings realized by meta-mesh design are due to meta-mesh's ability to bypass chains. While it was not confirmed in any test cases here, it is possible, however, that reduction in spare capacity requirements on some chains can make new working routes slightly more attractive than in a conventional span restoration design, accounting for some of the differences observed.

Of particular interest is how closely the capacity cost reduction curves are paralleled by the percentage express flow curve in each figure. While there are some notable differences in their behaviours, it is quite clear that test cases with a high amount of express flow tend to be the same test cases where capacity savings are the greatest, which is consistent with and validates our understanding of how meta-mesh functions. When working routes fully transiting a chain are allowed to use its chain bypass instead, there is a corresponding decrease in spare capacity requirements (and no accompanying increase in wavelength-kilometres of working capacity). The greater the amount of working routing making use of bypass spans, the greater the capacity savings should be.

Qualitatively, we can observe that capacity cost savings (as well as express flow working capacity) generally appear to increase somewhat for test case networks with low average nodal degree. For some network families, there is also a perceptible drop in capacity cost savings in test cases at the extreme low end of the range of $\bar{d}$ values. This latter behaviour where savings first increase as $\bar{d}$ is lowered but then drop off again toward the most sparse cases is particularly apparent in the 15n30s1 network family and was also observed in a 32-node network family studied in [25] and [54]. A reasonable explanation becomes clear when we consider the following. As $\bar{d}$ decreases, the number and length of chains increases, and so more and more working capacity is used by express flow. However, at some intermediate $\bar{d}$, a continued reduction in the number of spans will cause more nodes (which were previously sources or sinks of express flow) to become a part of a chain, thereby converting their demands into intra-chain demands from the point of view of those chains. Thus, a continued reduction in $\bar{d}$ will diminish meta-mesh restoration's ability to make use of chain bypasses and the spare capacity savings that result. Indeed, in the logical limit of a Hamiltonian cycle, all demands are intra-chain, and there can be no express flow whatsoever, and therefore no spare capacity savings in the sense pursued here. The relative benefit of meta-mesh restoration is also small for highly connected networks because they contain few chains, if any, and the meta-mesh and conventional designs are then identical.

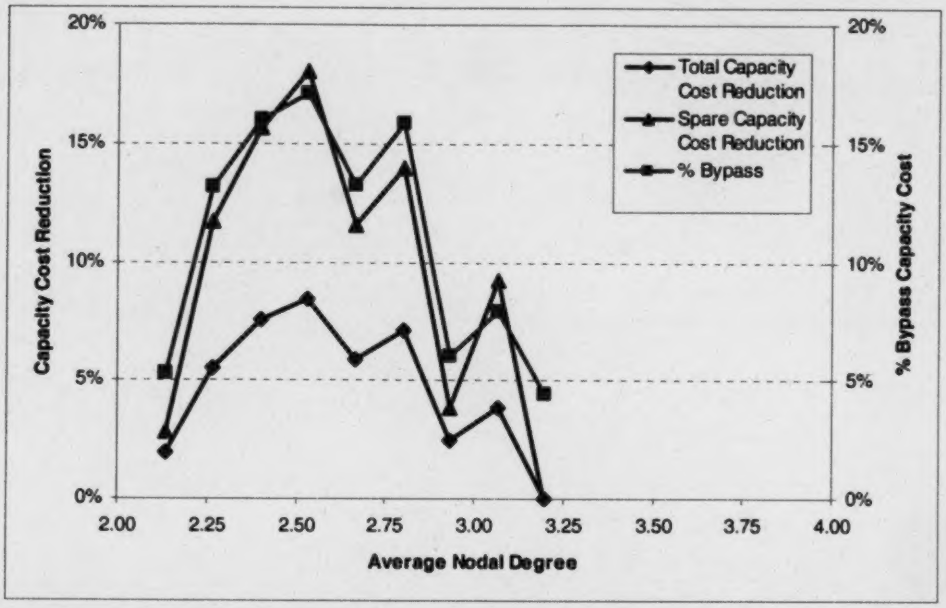


Figure 7.25 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 15n30s1 network family.

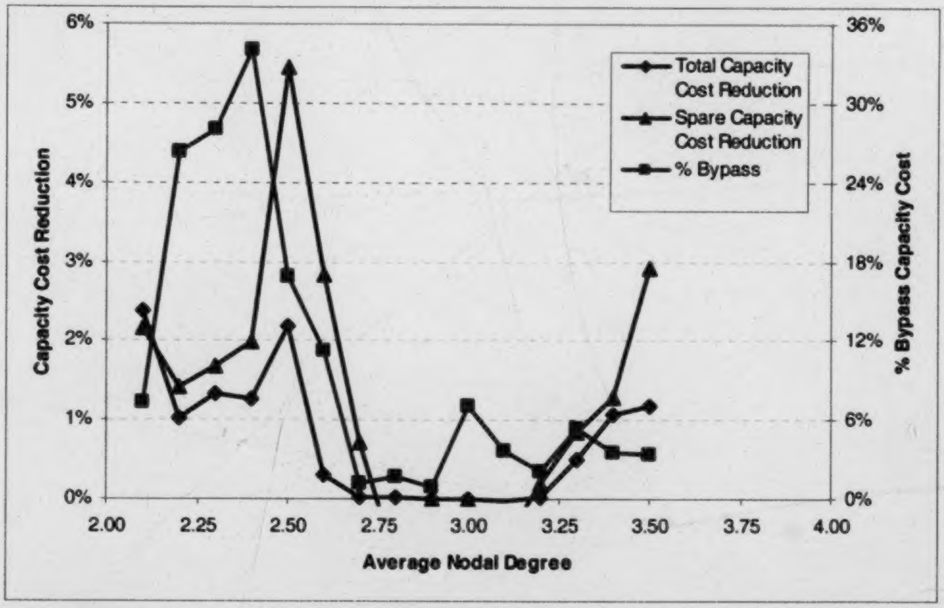


Figure 7.26 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 20n40s1 network family.

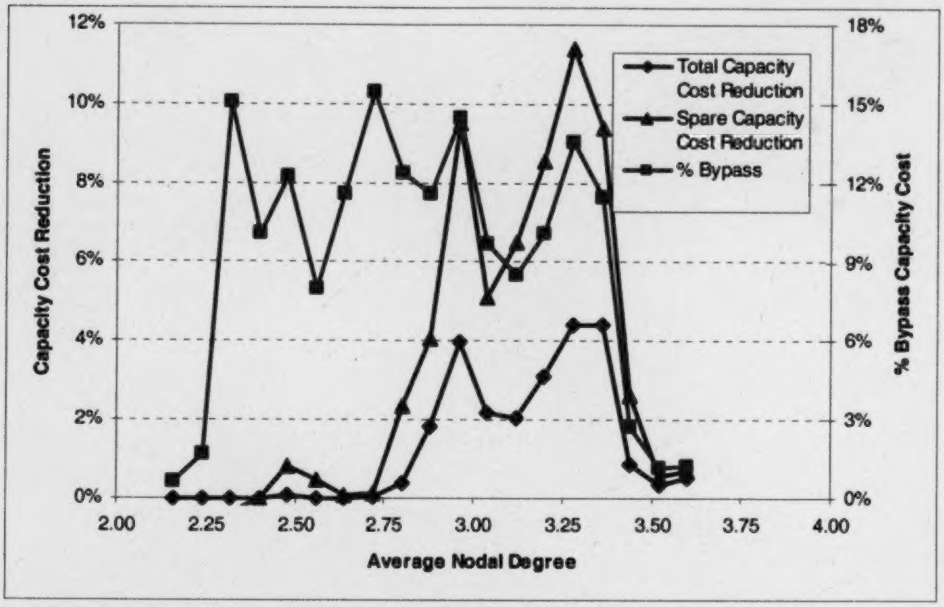


Figure 7.27 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 25n50s1 network family.

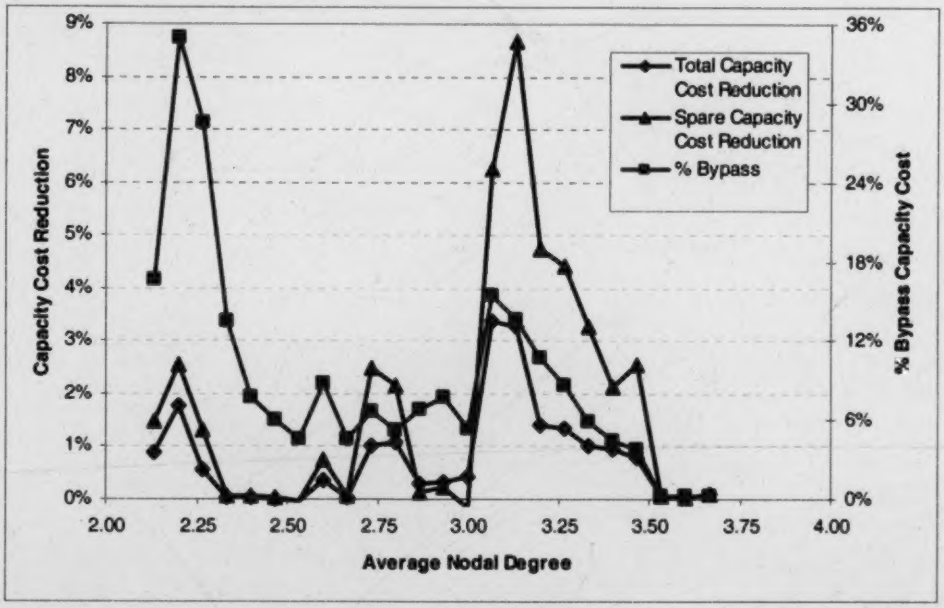


Figure 7.28 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 30n60s1 network family.

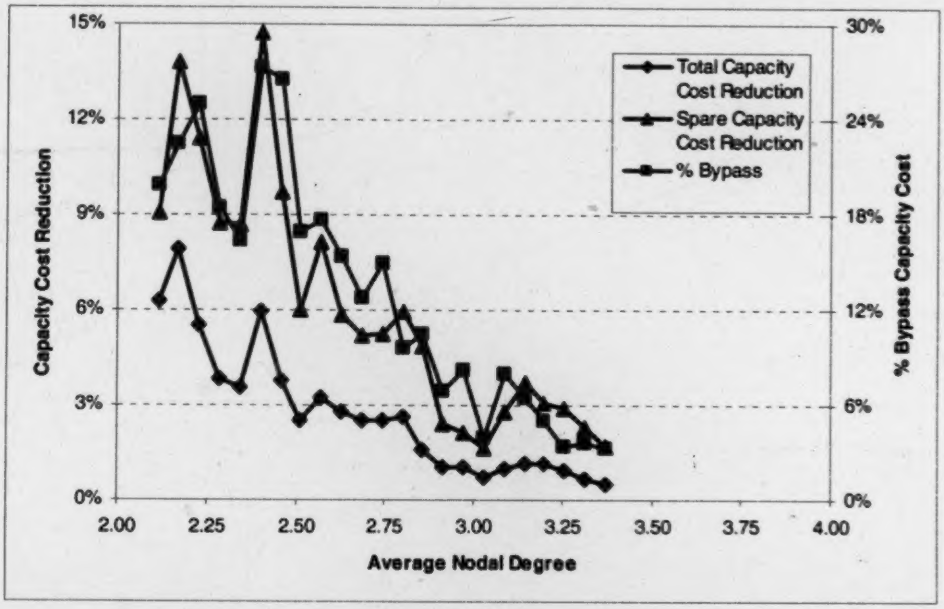


Figure 7.29 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 35n70s1 network family.

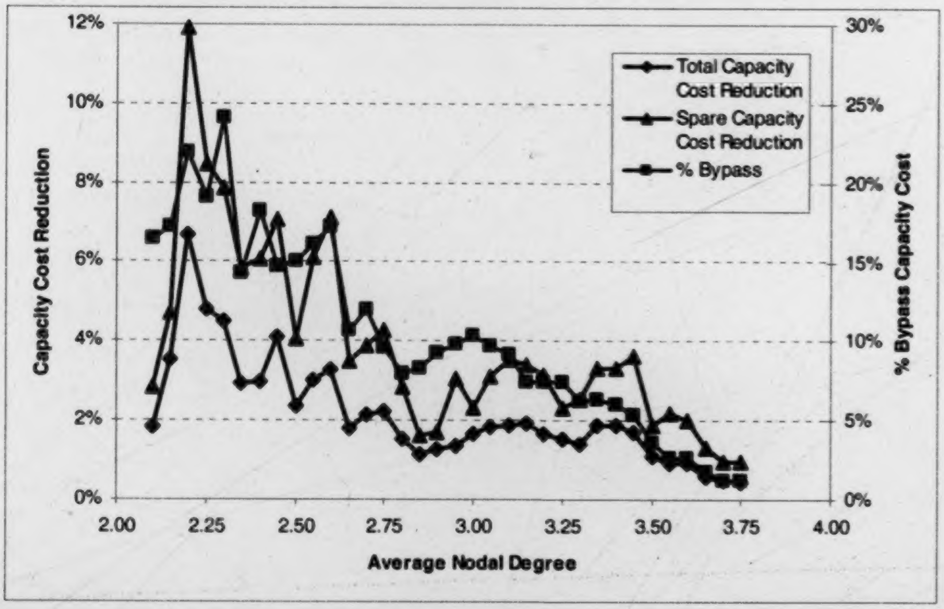


Figure 7.30 – Meta-mesh total capacity cost reduction relative to span restoration JCA and bypass span usage for the 40n80s1 network family.

In Figure 7.31 and Figure 7.32, we return to the idea that capacity cost savings in meta-mesh are related to a network's average nodal degree, and are to some extent proportional to the amount of express flow in the network's chains. In both figures, each data point represents either the spare or total capacity cost savings (as indicated) of an optimally designed meta-mesh network relative to an optimally JCA-designed span-restorable network plotted against average nodal degree (in Figure 7.31) and against the percentage of express flow in the network (in Figure 7.32). To better illustrate any relationship between capacity cost savings and $\bar{d}$ or express flow, we have superimposed trend lines over the scatter plot. The trend lines correspond to a *simple estimated linear regression* of the data, where *method of least squares* is used to estimate the intercept and slope of the regression line [8], [85], [116]. The regression line is estimated by:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \cdot x \tag{7.7}$$

where $\hat{y}$ is the estimated value of the *response variable* (e.g., the spare capacity cost savings) in a network with the specified *predictor variable*, x (e.g., the average nodal degree), $\hat{\beta}_0$ is the estimated intercept of the regression line (i.e., the value of $\hat{y}$ at $x = 0$), and $\hat{\beta}_1$ is the estimated slope of the line. $\hat{\beta}_1$ and $\hat{\beta}_0$ are calculated by the least squares estimate equations (7.8) and (7.9), respectively, where $\bar{y}$ and $\bar{x}$ are the averages of the respective data points, and n is the number of such data points [85].

$$\hat{\beta}_1 = \frac{S_{xy}}{S_{xx}} = \frac{\sum_{i \in \{1 \ldots n\}} y_i \cdot (x_i - \bar{x})^2}{\sum_{i \in \{1 \ldots n\}} (x_i - \bar{x})^2} \tag{7.8}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \cdot \bar{x} \tag{7.9}$$

The *coefficient of determination*, R^2, is often used to judge the accuracy of an estimated regression line [85], and is calculated by equation (7.10). R^2 can be thought of as the amount of variability in the data that can be accounted for by the estimated regression line, and is always $0 \le R^2 \le 1$. In general, the closer to $R^2 = 1$, the better the regression model, but R^2 is only an indicator of a model's appropriateness, and large values of R^2 do not necessarily imply a very accurate prediction of the data. A high R^2 is, however, a reasonably good indication that the estimated regression line is in the broad vicinity of the true relationship that is modelled [85].

$$R^2 = 1 - \frac{SS_E}{SS_T} = \frac{\sum_{i \in \{1 \ldots n\}} (y_i - \hat{y}_i)^2}{\sum_{i \in \{1 \ldots n\}} (y_i - \bar{y})^2} \tag{7.10}$$

The estimated regression line equations and their R^2 values are also shown in Figure 7.31 and Figure 7.32. In each figure, the upper equation and R^2 value (in green font) correspond to the spare capacity cost savings data, and the lower equation and R^2 value (in blue font) correspond to the total capacity cost savings data. As we can see in Figure 7.31, there appears to be a slight relationship between meta-mesh capacity cost savings and average nodal degree, as the data points in the scatter plot seem to orient somewhat from the lower right of the plot (high $\bar{d}$ and low capacity cost savings) to the broadly spread out upper left area (low $\bar{d}$ and higher capacity cost savings). Similarly, in Figure 7.32 the data points in the scatter plot appear to be oriented somewhat from the lower left region of the plot (low percentage of express flow capacity and low capacity cost savings) to the upper right region (high amounts of express flow and higher savings). However, we can see from the relatively low R^2 values of the respective trend lines, particularly those for the average nodal degree, that the data points cannot be strongly matched to the trend line equations given. Nonetheless, we can observe that meta-mesh capacity cost savings do appear to have a greater linear relationship to the percentage of express flow in a network than the network's absolute connectivity.

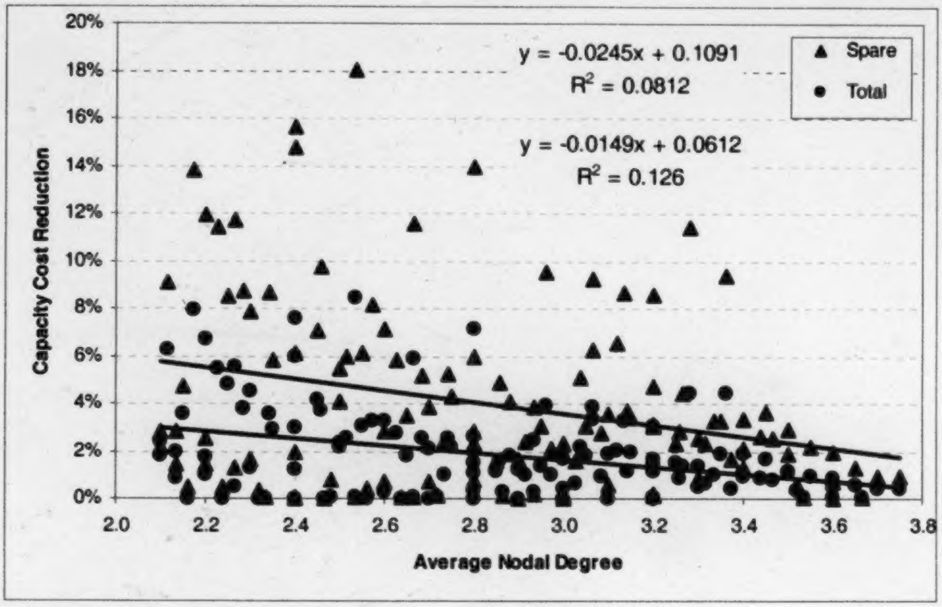


Figure 7.31 – Meta-mesh total capacity cost reduction relative to span restoration JCA versus average nodal degree for all 163 test case networks.

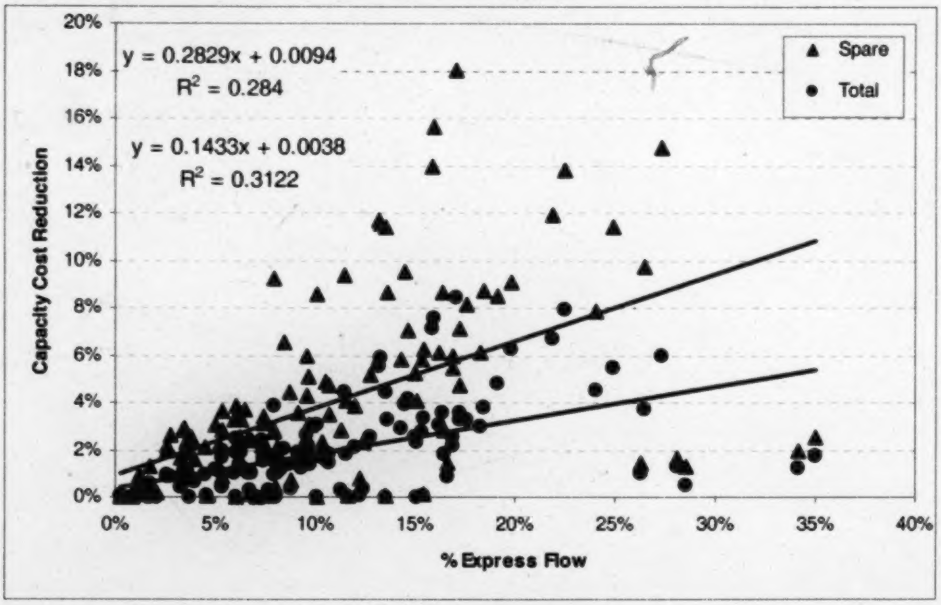


Figure 7.32 – Meta-mesh total capacity cost reduction relative to span restoration JCA versus express flow capacity for all 163 test case networks.

However, in lieu of any accurate estimation of the relationship between capacity cost savings and average nodal degree or the amount of express flow, we can also calculate the *sample correlation coefficient*, r_{xy}, [8], [85], where r_{xy} is calculated by equation (7.11).

$$r_{xy} = \frac{s_{xy}^2}{s_x \cdot s_y} \tag{7.11}$$

s_{xy}^2 is the *sample covariance* between the variables, while s_x and s_y are the sample standard deviations of the two variables. Their values are determined by equations (7.12), (7.13), and (7.14), respectively.

$$s_{xy}^2 = \frac{\sum_{i \in \{1 \dots n\}} (x_i - \bar{x}) \cdot (y_i - \bar{y})}{(n-1)} \tag{7.12}$$

$$s_x = \sqrt{\frac{\sum_{i \in \{1 \dots n\}} (x_i - \bar{x})^2}{(n-1)}} \tag{7.13}$$

$$s_y = \sqrt{\frac{\sum_{i \in \{1 \dots n\}} (y_i - \bar{y})^2}{(n-1)}} \tag{7.14}$$

For any pair of predictor and response variables, *x* and *y*, $-1 \le r_{xy} \le 1$, where $r_{xy} \approx 0$ implies that *x* and *y* are independent of each other. If, on the other hand, $r_{xy} \ne 0$, then *x* and *y* are said to be *correlated*. A positive correlation, or $r_{xy} > 0$, implies a linear relationship with a positive slope (i.e., large values of *x* correspond to large values of *y*), while a negative correlation, or $r_{xy} < 0$, implies a relationship with a negative slope (i.e., large values of *x* correspond to small values of *y*). The proximity of r_{xy} to either extreme provides a measure of the linear relationship between *x* and *y*, where the larger the absolute value of r_{xy}, the more closely correlated the two variables are.

Calculations of the sample correlation coefficients of spare and total capacity cost savings with $\bar{d}$ and the percentage of express flow in a network also support the observation that they are at least loosely linearly correlated. For spare and total capacity cost savings crossed with $\bar{d}$, we find $r_{xy} = -0.275$ and $r_{xy} = -0.355$, respectively, which correspond to the negative slopes of the trend lines in Figure 7.31. For spare and total capacity cost savings crossed with the percentage of express flow in a network, we find $r_{xy} = 0.533$ and $r_{xy} = 0.559$, respectively, which correspond to the positive slopes of the trend lines in Figure 7.32. The fact that the latter two correlation coefficients have larger absolute values is related to the observation that the data points in Figure 7.32 are more closely clustered near their trend lines than those of Figure 7.31. In other words, the amount of express flow in a network has a greater impact on meta-mesh capacity cost savings than a network's connectivity. This serves to validate our understanding of how and when the meta-mesh design strategy works: the greater the amount of express flow over chains, the greater the opportunity to eliminate loop-back spare capacity requirements within them. It is also compatible with the observations of the capacity cost savings curves in Figure 7.25 through Figure 7.30, which only truly appear to follow a linear relationship with $\bar{d}$ for the 35n70s1 and 40n80s1 network families.

The reason the correlation coefficients of capacity cost savings versus percentage of express flow aren't larger is likely because of the complex nature of network capacity design. The total capacity cost of any optimally designed network is related to quite a wide variety of factors, particularly the minute details of the network's topology (including much more than $\bar{d}$ and the amount of express flow through the network's chains). Any of those factors can have quite unpredictable effects on the overall design itself, which introduces a considerable amount of variability in the data points of Figure 7.31 and Figure 7.32. The number of chain spans in the network, as well as the percentage of all spans in a network that are part of a chain for instance, are two other factors we can relate to the meta-mesh capacity cost savings relative to span restoration.

In Figure 7.33 and Figure 7.34, we plot capacity cost savings against the actual number of chain spans in a network and the percentage of spans in a network that are part of a chain, respectively. As further validation that the basis for meta-mesh capacity reductions is associated with the ability to reduce the amount of loop-back spare capacity within chains, the scatter plots of the data and their trend lines indicate a linear relationship nearly as strongly as that in Figure 7.32. For the data in Figure 7.33, we calculate $r_{xy} = 0.384$ and $r_{xy} = 0.453$, for the spare and total capacity cost savings, respectively, and the data in Figure 7.34 give $r_{xy} = 0.447$ and $r_{xy} = 0.509$.

Another calculation we can make is the correlation coefficient between the percentage of express flow and the percentage of spans in the network that are part of a chain. This yields $r_{xy} = 0.818$, which suggests a very strong linear relationship between the two parameters. The reason for the very strong correlation is quite clear: the amount of working flow routed fully through chain spans must be related at least in part to the number and length of such chains in a network. A similar analysis of the percentage of express flow in relation to the average nodal degree of the network also indicates a relationship but not as strongly, with $r_{xy} = -0.678$.

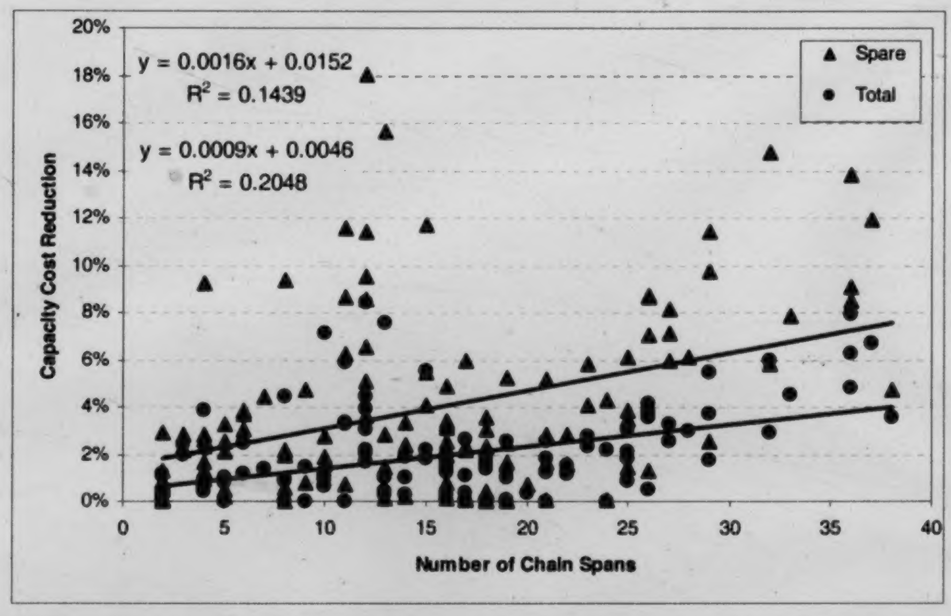


Figure 7.33 – Meta-mesh total capacity cost reduction relative to span restoration JCA versus the number of chain spans for all 163 test case networks.

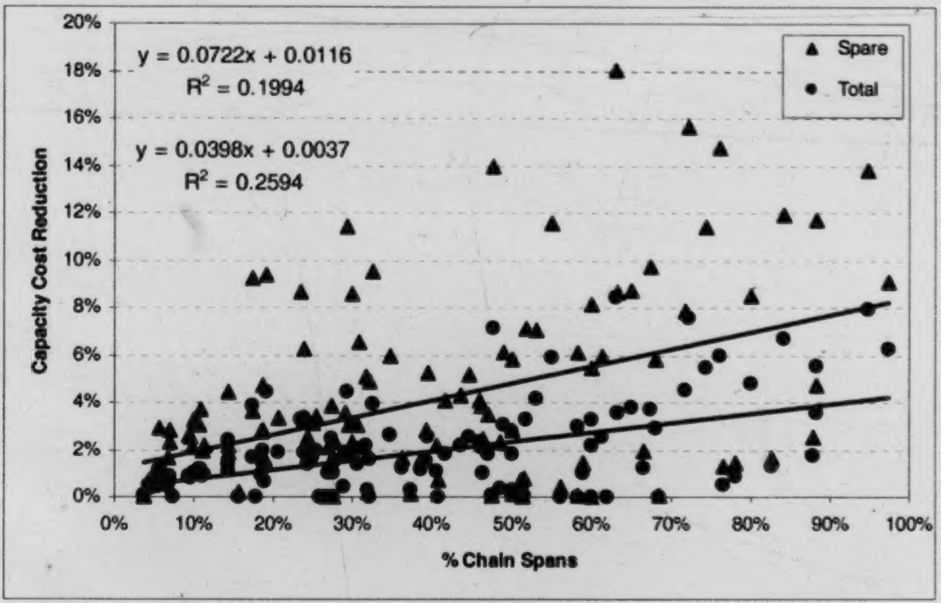


Figure 7.34 – Meta-mesh total capacity cost reduction relative to span restoration JCA versus the percentage of chain spans for all 163 test case networks.

7.6.2.2 LOGICAL CHANNEL COUNT REDUCTION

In addition to the direct capacity cost savings provided by meta-mesh restoration, there is an accompanying decrease in *logical channel counts* as well. While the costs for ducts, fibres, amplifiers, regenerators, etc., scale with wavelength-kilometres of capacity, certain nodal termination costs on OXCs and OADMs scale only with channel counts, and are not distance-dependent (a logical channel is terminated on each end, regardless of distance, by either an OXC or OADM). We ordinarily model a network's cost as directly proportional to the wavelength-kilometres of working and spare capacity, and disregard any termination costs or any other such per-channel costs, and as long as span lengths do not vary too widely, this sort of approximation is acceptable. However, meta-mesh restoration introduces the possibility of additional savings beyond the conventional wavelength-distance costs we normally consider.

Two options discussed in Section 7.3.1 to carry out meta-mesh restoration was to route express flows through straight fibre splices or glass-throughs at each node within a chain, or even on separate fibres directly connecting the chain's anchor nodes. Either would allow express flows to physically bypass OADMs within the chain altogether. Any costs once associated with the express flow being terminated and handled by OADMs within the chain are therefore eliminated, and given sufficient express flow on some chains, entire OADMs could even be decommissioned and/or reused elsewhere within the network.

Figure 7.35 through Figure 7.40 show a breakdown of the working, spare, and total logical channel count savings in optimal lowest-cost meta-mesh designs relative to optimal span restoration JCA designs. Each figure provides data for a single network family, with data points divided into three curves, one each for working channel count savings, spare channel count savings, and total (working plus spare) channel count savings. Each data point represents the channel count savings of the optimal solution for the member of the family with the indicated average nodal degree, $\bar{d}$. The meta-mesh spare capacity cost reduction (relative to span restoration JCA) has been added to each figure for reference and comparison purposes (the solid black curve scaled to the secondary y-axis on the right).

As would be expected, the reduction in spare logical channel counts (the red curves) is almost perfectly paralleled by the reduction in spare capacity (solid black curve) in all network families. Calculation of the correlation coefficient between reductions in spare capacity cost and spare channels yields $r_{xy} = 0.966$, and a regression analysis provides an estimated regression line of $y = 0.8869x - 0.001$ (y is the spare capacity cost reduction and x is the spare channel count reduction) with $R^2 = 0.933$, all of which supports that observation.

In virtually all test cases, meta-mesh provides a substantial reduction in *working* logical channel counts as well, despite the fact that working capacity cost reductions were essentially non-existent (see Section 7.6.2.1). On average, working logical channel counts were reduced by 6.2% over all test cases, and in 22 of the networks, the reductions were in excess of 10%, with a best-case reduction of 27.9% in network 30n60s1-33s. Further analyses confirm that working logical channel count reductions are very strongly linearly correlated to the percentage of express flow over a network's chains, with a correlation coefficient of $r_{xy} = 0.941$ and an estimated regression line of $y = 0.6156x - 0.0022$ (y is the working channel reduction and x is the percentage of express flow) with $R^2 = 0.885$. Clearly, the reduction in working logical channels is primarily due to express flow now being routed over logical chain bypasses, rather than over the chains themselves. In general, working channel reductions were even greater than spare channel reductions, which were an average of 4.5% over all test cases. Total logical channel count reductions were an average of 5.2% over all test case networks, and in the best case, network 40n80s1-44s had a 16.1% reduction in total logical channel counts.

The obvious benefit is that, as discussed above, meta-mesh will allow a network operator to further reduce costs by the elimination of a significant portion of link termination costs, and could even allow removal of some intra-chain OADMs. Allowing the design formulation to consider logical channel counts in its optimization could even enhance this effect beyond what is already inherent in the model. A simple modification of the objective function in equation (7.1) to the in equation (7.15), or perhaps equation (7.16) would suffice. Here, α is a trade-off parameter whose value would be under the network planner's control [27], and would depend in part on the relative expense of termination costs and distance-dependent capacity costs, and the planner's desire to bias the model towards one or the other. If α is small, the network design would not differ greatly from the basic meta-mesh design. However, if α is large, the resultant network would be more biased towards reducing logical channel counts rather than capacity costs in the conventional sense.

$$\text{Minimize} \quad \sum_{\forall j \in S} \left(c_j \cdot \left(s_j + w_j \right) + \alpha \cdot w_j \right) \tag{7.15}$$

$$\text{Minimize} \quad \sum_{\forall j \in S} \left(c_j \cdot \left(s_j + w_j \right) + \alpha \cdot \left(s_j + w_j \right) \right) \tag{7.16}$$

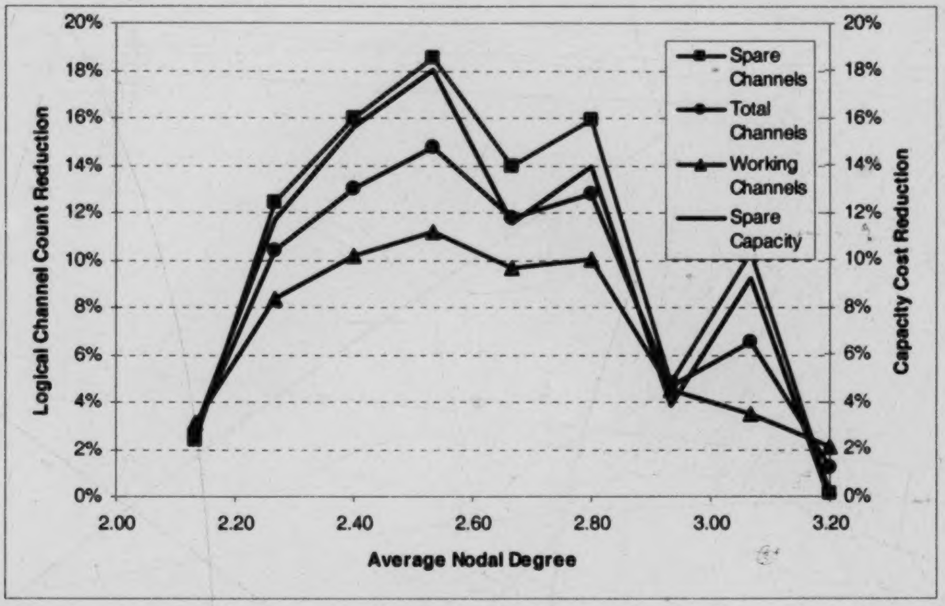


Figure 7.35 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 15n30s1 network family.

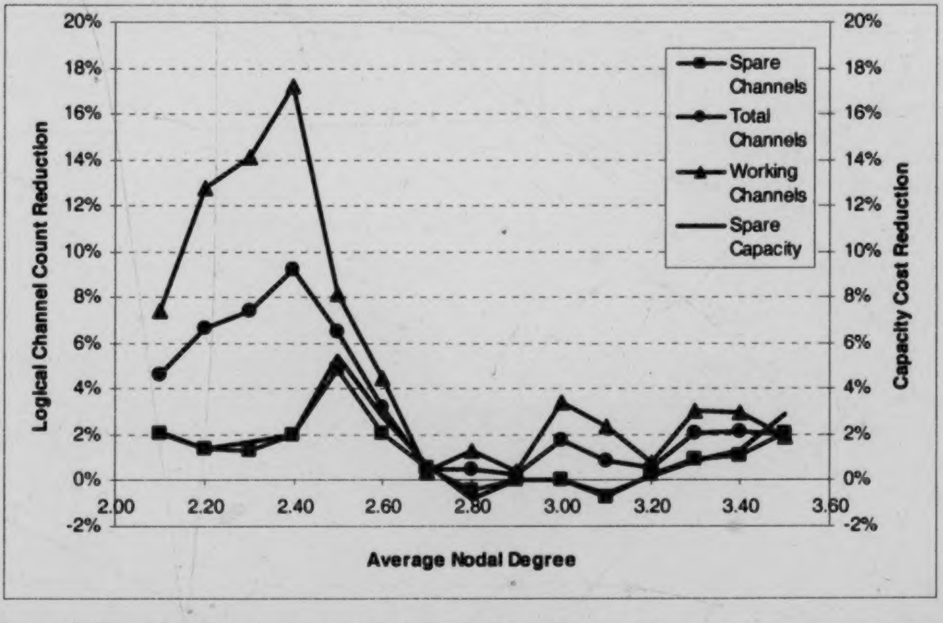


Figure 7.36 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 20n40s1 network family.

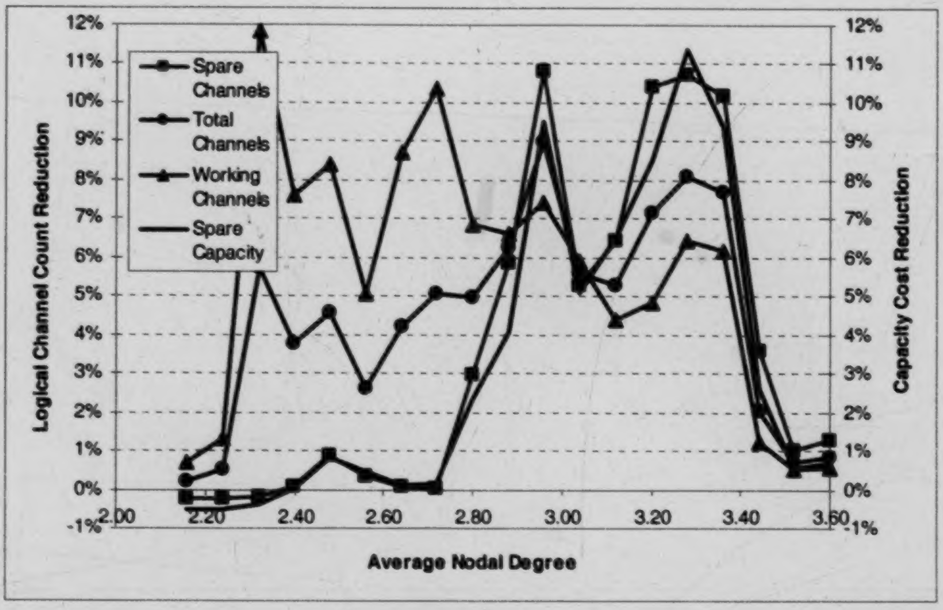


Figure 7.37 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 25n50s1 network family.

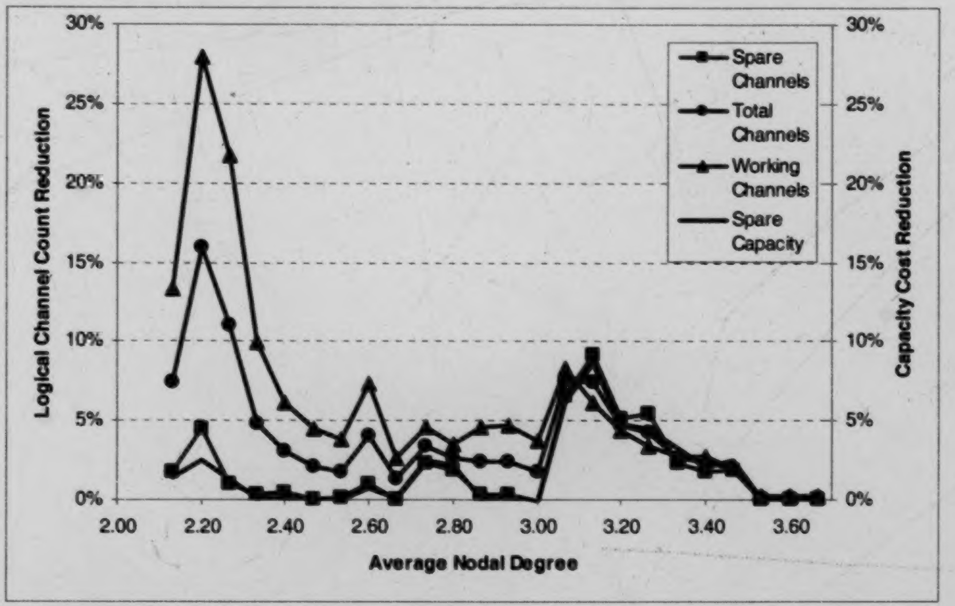


Figure 7.38 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 30n60s1 network family.

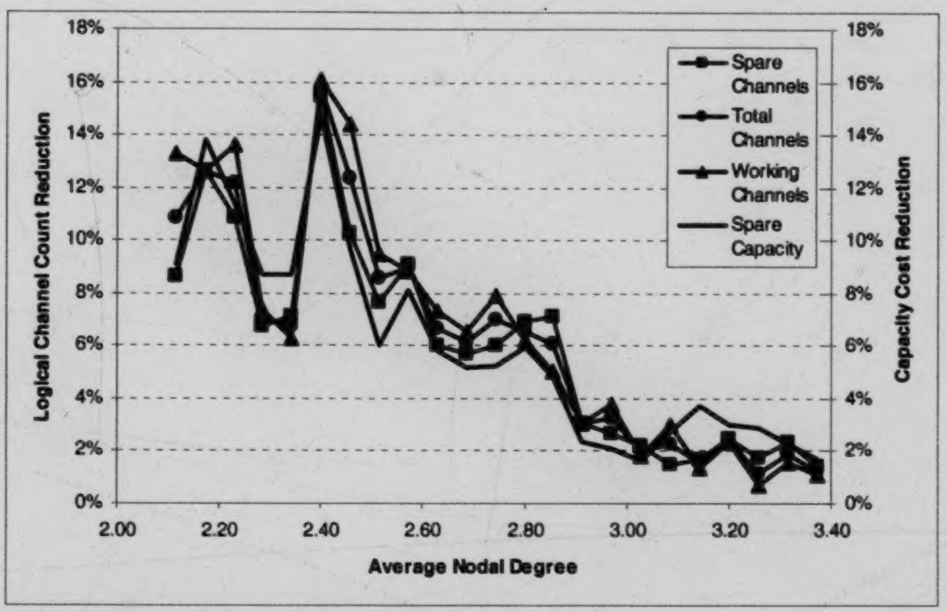


Figure 7.39 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 35n70s1 network family.

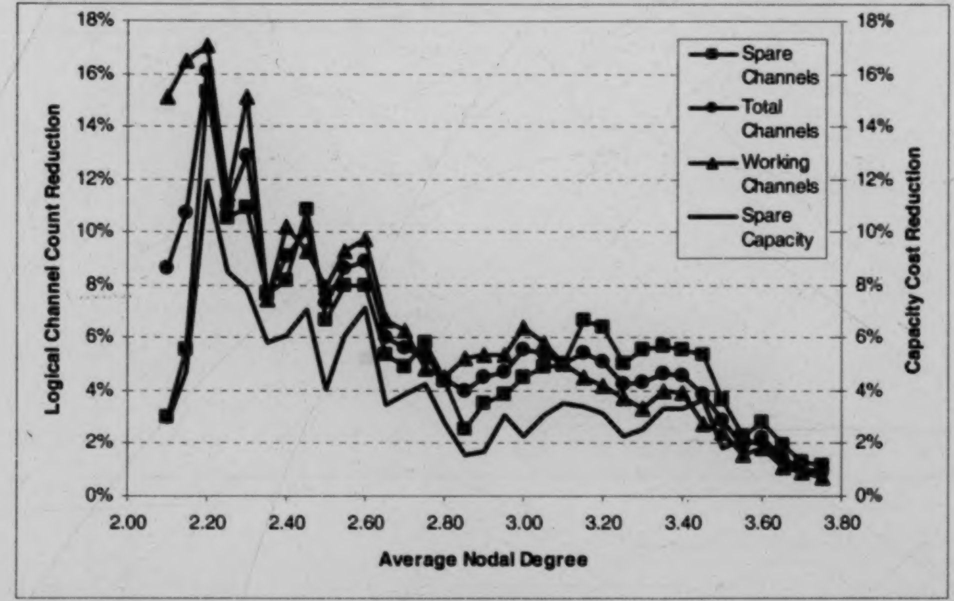


Figure 7.40 – Breakdown of meta-mesh logical channel reductions relative to span restoration JCA for the 40n80s1 network family.

7.7 CLOSING REMARKS

A close look at the makeup of working traffic in a sparse network with chains of degree-2 nodes will show that a portion of the working flow traversing a chain is actually express flow from wavelengths that fully transit the chain end-to-end. Intra-chain or local flow from wavelengths terminating/originating in the chain requires loop-back restoration since their aggregate composition is modified by add/drop actions within the chain. We've shown, however, that express wavelengths do not. Rather, they can be restored by allowing them to fail all the way back to the anchor nodes of the chain, which effectively allows them to be treated entirely within the more highly connected meta-mesh of the network graph topology, thereby decreasing spare capacity costs and redundancy.

Implementation can use a straight fibre splice or glass-through at each OADM within the chain for express wavelength traffic (presumably, OXCs are not needed since we're dealing with degree-2 nodes). Upon failure of a chain span, failure of local wavelengths is detected by the OADMs on either side of the break (which then initiate the loop-back mechanism as far back as the anchor node OXCs), while failure of the express wavelengths is detected by the anchor node OXCs. Both types of failed traffic are then logically unified for restoration between the OXCs at the anchor nodes on spare capacity distributed throughout the remainder of the network.

Using the benchmark span restoration JCA ILP model as a starting point, we developed a new ILP formulation that models our proposed meta-mesh restoration technique. Experimental results showed that the meta-mesh design model was able to provide an improvement in network design costs relative to the benchmark. The interesting and initially surprising overall trend is that the reduction in capacity requirements tended to peak in networks with low to intermediate average nodal degrees. In networks with higher or very low average nodal degrees, we see a drop in the benefit gained by meta-mesh restoration (i.e., the reduction in capacity costs is less). Further analysis showed that a more important factor in predicting capacity cost savings provided by meta-mesh restoration was the percentage of a network's spans that are part of a chain. The meta-mesh model produces its benefits by allowing express flows through chains to fail back to the chains' anchor nodes. The benefit or contrast relative to the benchmark model is then greatest when there are many long chains being traversed by a significant number of express-flow wavelengths. The more sparse a network gets, the greater the number of wavelengths that have to transit chains to reach their destinations. But a continued reduction in average nodal degree eventually reaches a point where the network is composed almost entirely of a few large chains. Wavelengths that once transited chains end-to-end progressively more frequently find their sources and destinations within chains themselves and become intra-chain wavelengths, and there can be no express-flow related savings in the sense pursued here. On the other hand, when the network is very richly connected, there are also very few express-flow wavelengths (although this is because these networks contain few, if any, chains), and the conventional benchmark design model cannot be improved upon.

We also see that there is a corresponding, albeit larger, reduction in constituent logical wavelength channel counts as well, due to the routing of express flows over logical chain bypass spans. Working channel counts were reduced as much as 27.9% in the best case, and by 6.2% on average, providing additional savings to the network operator beyond the conventional distance-weighted capacity costs since OADM termination costs within the chain can be significantly reduced. Given sufficient decreases in such channel counts, elimination of chain OADMs might even be possible.

7.7.1 FUTURE DIRECTIONS

Several obvious opportunities for further investigation present themselves in this work. The first is to implement the modified ILP formulation discussed at the end of Section 7.6.2.2, and quantify the effects of introducing an additional incentive to reduce logical channel counts, primarily working channels. With properly determined values of α to reflect real termination costs, resultant network designs will likely see even more significant cost savings over conventional span-restorable designs.

A second opportunity for future work is to exploit the possibility of eliminating intra-chain OADMs when the core bandwidth they're required to handle/terminate is reduced. A modified ILP formulation could be developed to include a stepwise capacity cost model to characterize the modular aspect of installing/decommissioning individual pieces of nodal equipment.

Another interesting question to probe is whether an existing span-restorable network could accommodate a significant increase in demand by implementing meta-mesh restoration. We would expect that demand increases in the order of the capacity cost savings already observed would likely be possible. The conversion of intra-chain OXCs, if any, into OADMs, as well as the elimination of some intra-chain OADMs could also be incorporated into a design method to provide real cost savings in parallel with the capability to handle greater amounts of traffic demands.

Finally, in the event that physically distinct bypass spans (say, using glass-throughs or separate fibres) are not employed, complete development of a meta-mesh protocol, say, based on SHN, [45] and [47], would also provide an interesting area of further study. The protocol would need to incorporate an efficient means of identifying which wavelengths are to be failed all the way back to the anchor nodes, and a procedure for logically unifying the restoration process at the anchor nodes to accommodate the local and express wavelengths affected. Implementation within a network modeling software package such as OPNET or perhaps within an experimental testbed network such as TRnet [112] would allow proper characterization of the protocol's operation.

CHAPTER 8

TOPOLOGICAL TRANSPORT NETWORK DESIGN[12]

In virtually all of the optimization problems so far posed on mesh-restorable networks, and all those so far discussed herein, the graph topology of the physical facility routes is known and given. In practice most facilities-based network operators entered the current era with a legacy topology or a pre-determined topology arising from a prior railway or gas-pipeline utility company right-of-way structure. Traditionally, new spans or edges in the facilities graph topology would be added as needed on a case-by-case basis, and driven more by the economics of working demand conveyance than from a standpoint of restoration capacity sharing or global topology optimization. Before about 1985 and the widespread deployment of fibre optics, which was quickly followed by an urgent need for restoration, many long-haul networks were tree-like, optimized solely to serve the working demands with no concern for network-level restoration. In those days, tree-like topologies were more viable because of the greater reliance on digital microwave radio systems, which have a high inherent availability. On the other hand, today's fibre-based transport relies on cables, which have much lower structural availability. Closed (two-connected or biconnected) network graph topologies and active restoration schemes have therefore become essential to the widespread deployment of fibre optic transport systems.

Unlike the case in private leased-line network design where any desired point-to-point *logical* edge can be provided for a virtual network, it is usually quite difficult and very expensive to augment the topology of the underlying *physical* facilities graph. Consequently the topology of some of today's facilities-based network operators tends to comprise a tree-like pre-1990s topology that was simply closed or made biconnected in the most expeditious manner so that fibre rings would be feasible. More recent entrants have topologies arising almost wholly from prior utility infrastructures. In either case, however, the network topologies were not optimized from a global topological standpoint.

---

[12] This chapter contains some material previously published in [51] and [53].

It is well known that even small changes in the physical topology of a network can have quite significant effects on the amount of working and spare capacity required to route and protect all of the network's demands. We saw in Section 6.4 that introducing just one new span to the network topology will allow more efficient sharing of capacity (and therefore decreased spare capacity costs), but the extent to which this is so will depend on exactly where that span has been added. An important issue for network operators today is therefore how to optimally evolve their physical network topologies, or even how to optimally design a completely new network topology such that costs are minimized. So the natural next step in survivable network design research is to incorporate the design of the physical graph topology as yet another output of the optimization problem, allowing the working and restoration routing and spare capacity sharing to drive the topology, rather than the other way around.

In this chapter of the thesis, we address the complete *green-fields* network design problem, where no physical infrastructure yet exists. The green-fields case lends itself best to overall insights about the problem and has the most generality as the canonical research model. In practice, there would more often than not be some established set of edges and perhaps only a short list of possible new route acquisitions for incremental topology growth or evolution. We will also address corresponding methods for incorporating any pre-existing infrastructure in the design problem.

Each candidate edge in the topology design problem can be thought of as representing a facilities right of way over which an essentially unlimited capacity of transmission systems can be installed to support working and restoration routing requirements. A one-time fixed cost is incurred for the acquisition and preparation of any new edge in the physical facilities graph, followed by perhaps multiple levels of stepwise incremental costs dependant on the capacity of the installed systems. Both the fixed and incremental capacity costs are distance dependant in the general case since the fixed cost includes rights-of-way and lease acquisitions, excavation, duct installation, equipment housing every 50 km for amplification, etc., while incremental capacity costs incorporate all per-channel costs such as requirements for additional fibre, wavelength channel terminations, amplifiers, etc. There could even be a coarse step-wise incremental cost associated with fibre transmission systems of various sizes or the lighting up of new fibre pairs with an initial block of DWDM carrier wavelengths, as well as finer-scale incremental costs as additional individual wavelength channels are lit within each system to provision new services. To simplify formulation of the problem, however, we consider only a single cost structure to approximate reality, but modularity-type adjustments to the formulations developed could be used to model any number of cost structures.

The complete problem includes the simultaneous selection of a set of edges that comprises a closed connected graph, the routing and provisioning of capacity for working flows, and the provisioning of restoration routes and spare capacity, so that the network serves all demands and is fully restorable[13] against any single edge failure, all at minimum total cost. We refer to those three main aspects of the problem as topology, routing, and sparing, respectively. As will be seen, the computational complexity of solving the complete problem is practically overwhelming for all but small instances. Topology and routing alone constitute a multi-commodity instance of the *fixed charge plus routing* problem [39], [73], which is a notoriously difficult NP-hard problem in its own right. But the full problem also involves the simultaneous optimization of restoration routing and spare capacity allocation, which by itself is an NP-hard problem, even when the topology is fixed and known. These coupled subproblems have very different dependencies on graph topology. Solutions of the fixed charge plus routing problem tend towards sparse topologies like spanning trees, especially if the fixed cost of edges is high relative to the incremental cost of each channel. But those topologies are un-closed and inherently un-restorable by restoration re-routing. On the other hand, solutions for optimal spare capacity design are lower in cost when $\bar{d}$ is high, and all solution graphs have to be strictly closed.

In Figure 8.1, complete network design costs are shown for each member of the 40n80s1 network family (using the span restoration JCA design). In addition to the capacity costs already dealt with in prior chapters, a network design cost now includes the fixed establishment costs of all spans in the network's topology. The data in the figure is organized into three curves, one each for the capacity cost (in red), the topology cost (in blue), and the total cost (in green). Each data point represents the cost of the network at the indicated connectivity when optimally designed using the span-restorable JCA model. Topology costs are calculated by assigning a fixed establishment cost equivalent to 750 times the unit-capacity cost on each span. As already seen in CHAPTER 6, capacity costs decrease with increases in $\bar{d}$. Topology costs, on the other hand, increase with increases of $\bar{d}$, as we would expect; the more spans in the topology, the greater the associated cost. When those two components of the complete cost of a network design are added, we generally see a curve similar to the one in the figure, where a network is more costly at either high $\bar{d}$ or low $\bar{d}$, and least costly at some intermediate $\bar{d}$. Thus, the overall problem contains counteracting topological preferences that are linked under a min-cost objective for the determination of graph topology, working path routing, and restoration capacity placement. The goal is therefore to simultaneously determine a topology, a working path routing, and spare capacity design that will produce a network solution as close as possible to that minimum.

---

[13] Restoration will be assumed to be span restoration.

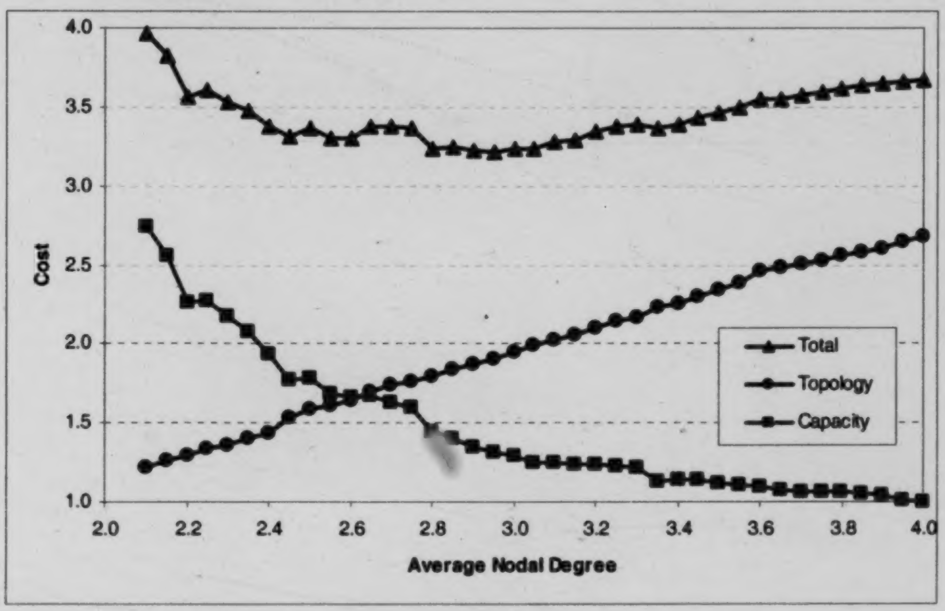


Figure 8.1 – Breakdown of sample complete network design costs for the 40n80s1 network family.

8.1 PRIOR WORK

Some prior work has addressed the topology design problem for communication networks to address access network design, expansion planning, wide-area packet data network design, and even backbone networking, but very little takes restoration concerns into consideration. An early paper dealing with survivable network design investigates various topological factors that allow a network to "survive an enemy attack or natural disaster" but stops short of actually developing design methods [36]. Studies in [7] and [22] investigate methods for designing tree-topologies (including link capacities) in backbone networks and data communication networks, respectively, but does not consider restoration or reliability in any way.

A genetic algorithm is used in [58] for topological design and dimensioning of an access network. The work only considers tree topologies (i.e., non-restorable networks), but develops an efficient means of representing topology genomes and assessing their fitness values. Some work in [71] studies topology optimization for multi-point line layout, where the requirement is for a minimum cost set of sub-trees rooted at a central node, and again, such networks are non-restorable.

In [82], a *branch exchange* heuristic[14] is used in a data communications network to seek new link additions to a spanning tree that maximize the ratio of the reduction in average delay to the increase in cost for the link. Branch exchange is also used to perform "topology tuning" to open new links to overcome congestion in an ATM network in [42]. It is pointed out in [71] that, while the basic branch exchange approach is quite general in nature and useful for a range of problems, its main drawback is that the re-routing of demands to evaluate the benefit at each step occurs within the inner loop of the process that generates the exchanges. [71] states that, "since routing itself is typically $O(N^3)$ this tends to make even simple branch exchange searches $O(N^5)$, which is prohibitive for moderate to large size networks." The performance of branch exchange heuristics can be improved by using cut saturation techniques as in [40]. The idea is that by detecting flow-saturated cuts of the graph (the minimal set of highly utilized spans that, if removed, would disconnect the highly utilized spans that, if removed, would disconnect the network), the branch exchange process can be guided to discover effective exchanges in fewer iterations. This is done by generating moves that remove a lightly loaded link from one side of the saturated cut and add a new span that crosses the cut. This effectively moves a lightly utilized capacity investment to increase the cross-section of the saturated cut. Heavily used cuts can be efficiently identified with a minimum spanning tree algorithm where link utilizations are used as the edge weights.

---

[14] Branch exchange is a class of heuristics that begins with a feasible topology and proceeds with local modifications such as addition, deletion, or exchange moves on graph edges, seeking to maximize some problem-specific figure of merit.

Topology design work in [10], [37], and [71] makes mention of mesh networking but they refer only the departure from pure tree topologies towards at least partially closed topologies. In those works, "mesh" simply refers to networks with more than a single route between node pairs rather than any implied capability of mesh restoration as we typically mean. They make no explicit requirements for restoration of any kind, but only recognise that closed networks are qualitatively more robust and so generally preferable on that basis alone. The work in [98] made similar observations and used the graph connectivity as a measure of its reliability, but there is no explicit survivability included in the design.

Aspects of topology design and network survivability were both addressed in [109] where another branch exchange heuristic approach is used to design an undirected network topology with specified redundancy. Here, survivability is not strictly assured but rather enhanced by providing specified numbers of disjoint paths between node pairs, with an understanding that such redundancies will "increase [the network's] survivability". In that study, costs for establishing new edges in the graph provide a basis for optimization, but there is no concept of capacity associated with an edge – the edge either exists or it does not. Work in [13] did consider both capacity assignment and topology design, but it does not explicitly consider survivability.

In [41], branch exchange is used in combination with an iterative process that alternates between flow determinations and capacity assignment. Here, they also include a network reliability constraint in their algorithm, but only as a verification step within the design phase to ensure that the topology obtained will remain connected with some specified probability given estimated link failure probabilities.

A widely used algorithm for data network topology design (MENTOR) is developed in [71] and [72]. It includes useful aspects of concentrator node location, but not survivability, and many topologies that result have "pendant nodes in the extremities of [a] tree". MENTOR is highly oriented to the issues of cost-versus-delay in data networking but it embodies some basic ideas of design strategy that may be useful in the restorable-mesh topology problem. Since it is quite efficient, it is potentially a very good procedure to be embedded within a heuristic for a more complete design problem or to suggest starting topologies for other methods. Any approach of the O(N^3) routing problem that involves consideration of all possible graph edges on N nodes must be O(N^5) or higher, but MENTOR, however, is O(N^2) and yet delivers good data network designs. The key is that MENTOR replaces the actual re-routing of demands with an easily computed surrogate criterion based on postulated hallmarks of a good routing solution. This allows MENTOR to skip a lot of the details in its basic iterations and look instead for general characteristics that are desirable from basic network design principles.

This philosophy is also found in the more recent Zoom-In approach [93]. Interesting ideas are presented for treating the sub-problems of topology, working routing, and restoration routing with surrogate problem abstractions and heuristics, followed by an exact optimization of routing and sparing on a fixed topology only when a final best topology is to be evaluated in detail. The Zoom-In approach uses a fast surrogate to approximate the sub-problems of demand routing and spare capacity assignment. Using a simple and fast surrogate for these sub-problems is evocative of the MENTOR philosophy and allows more topology options to be examined in the global search. The surrogate problem is to generate the capacity cost that corresponds to the "bi-routing" of each demand on two working routes. The demand matrix is first scaled up by a factor (1.2 is empirically suggested), and then half of each demand bundle is routed over the shortest path, with the other half routed over the shortest path that is link-disjoint from the first. The resulting total capacity is a representative upper bound on the cost of a detailed solution to the working capacity and sparing problem. With this process to evaluate the fitness of a proposed topology, a genetic algorithm is used to explore topology alternatives, with the surrogate problem solution representing the routing and sparing cost of the given topology in evaluating its fitness function. Once the GA on topology is completed, a detailed local optimization of the routing and sparing follows. The Zoom-In approach is also integrated into a multi-period network topology evolution method to generate several topological alternatives for each period [92]. This is followed by shortest path techniques to deduce which sequence of topologies offers a least cost network expansion plan over all time periods.

An ILP model for a combined topology and capacity design of a path-restorable network (without stub-release) is developed in [38]. Span selection is accomplished by use of modularity-type constraints and the "span-elimination" effects that accompany a joint modular design [24]. The cost model includes an installation charge as well as a capacity-related cost, but the capacity choices are restricted to a single "cable" of varying sizes on each span. Another drawback with this model is that it requires enumeration of eligible routes, and so will be quite difficult to solve for all but the smallest networks and only if candidate spans are limited to a subset of the complete graph of $N \cdot (N-1)/2$ possible spans.

8.1.1 FIXED CHARGE PLUS ROUTING

The problem of topology determination for minimum-cost edge selections plus routing costs has also been studied in the operations research community as the *fixed charge plus routing* (FCR) problem or alternatively the *fixed charge network flow* (FCNF) problem, [39], [73]. The communications network version is usually a multi-commodity problem where every O-D node pair may exchange non-zero demands. In its capacitated version it may have existing edge capacities and/or edge capacity limits to be respected. The FCR problem is equivalent to determining the topology whose combined edge selection costs and optimal working routing and capacity assignment costs are minimized.

Since we will build upon FCR in the work presented in this chapter, so we will cover it now in some detail. Recall from CHAPTER 5 that in addition to its arc-path form, we could also express an SCA or JCA design problem as a network flow or transhipment ILP problem [119], where supply/demand nodes act as sources/sinks of a lightpath demand commodity, and transhipment nodes act as intermediary points passing along the commodities to other nodes. While the arc-path formulation is effective and convenient for the problems seen so far, we have to abandon the arc-path method altogether in favour of a transhipment problem formulation to cope with the topology becoming part of the solution variables. Explicit enumeration of eligible working and restoration routes (as would be needed in an arc-path model) in an unspecified and potentially complete topology of $N\cdot(N-1)/2$ bi-directional edges becomes an intractable problem for even a small network, so the basic FCR problem is more easily expressed as a network flow problem.

The basic FCR ILP problem uses the following notation, some of which is repeated here from earlier notation used for other models:

Sets:

- *N* is the set of nodes in the network, and is typically indexed by *n*, *i*, or *j*, depending on the use. Despite the possible confusion that may result, we often also use *N* to represent the number of nodes in the network as well, and it is usually quite clear from the context which meaning is intended.
- *S* is the set of all possible directional edges in the network, and usually includes all $N\cdot(N-1)$ edges that are possible in the graph. It is typically indexed by the pair *i,j*, which represents the directional span from node *i* to node *j*. We technically treat edges in this model as being directional, but this is only to facilitate expression of the transhipment constraints. We still consider all spans, and therefore the flows they carry and the working capacity placed on them to be bi-directional, which corresponds to a real transport network.
- *D* is the set of all non-zero demand quantities exchanged between nodes, and is typically indexed by *r*. As will be seen later with definitions of origin and destination nodes, demands are treated in the model as being directional in the same manner as spans are above. Again, this is only to facilitate expression of the transhipment constraints, and we still consider demands, and therefore their flows to be bi-directional.

Input Parameters:

- d^r is the number of lightpath demand units for demand relation *r*.
- $O_r \in N$ is the node acting as the origin of lightpath demand *r*.
- $T_r \in N$ is the node acting as the target or destination of lightpath demand *r*.
- $c_{i,j} = c_{j,i}$ is the incremental cost of adding one unit of capacity to the edge in the graph between node *i* and node *j*. As mentioned above, directionality is implied in the model but bi-directionality is represented by asserting symmetry of the working capacity decision variables later.
- $F_{i,j}$ is the *fixed charge* or cost for establishment of an edge in the graph between node *i* and node *j*. It is sometimes also called the *edge cost* or *edge establishment cost*. Again, directionality is implied in the model but bi-directionality is represented by asserting symmetry of the edge decision variables later.
- *K* is an arbitrarily large positive constant, larger than any expected accumulation of working capacity on any one edge in the solution.

Decision Variables:

- $w^r_{i,j} \geq 0$ is the amount of working flow routed over the edge between nodes *i* and *j* (in the direction from *i* to *j*) for relation *r*.
- $w_{i,j} \geq 0$ is the integer amount of working capacity assigned to the directional edge between nodes *i* and *j* to support all working flows routed over that edge (in that direction).
- $\delta_{i,j} \in \{0,1\}$ is the 1/0 integer decision variable indicating whether the directional edge in the graph from node *i* to node *j* is to exist in the design. $\delta_{i,j} = 1$ if the edge is selected to exist, and $\delta_{i,j} = 0$ otherwise.

The FCR formulation itself is expressed as follows.

$$\text{Minimize} \quad \sum_{\forall i,j\in S}\left(c_{i,j}\cdot w_{i,j}+F_{i,j}\cdot\delta_{i,j}\right) \tag{8.1}$$

$$\text{Subject to:} \quad \sum_{\forall i,j\in S|i=n} w_{i,j}^{r}=d^{r} \qquad \forall r\in D \quad \forall n\in N\big|n=O_r \tag{8.2}$$

$$\sum_{\forall i,j\in S|j=n} w_{i,j}^{r}=0 \qquad \forall r\in D \quad \forall n\in N\big|n=O_r \tag{8.3}$$

$$\sum_{\forall i,j\in S|j=n} w_{i,j}^{r}=d^{r} \qquad \forall r\in D \quad \forall n\in N\big|n=T_r \tag{8.4}$$

$$\sum_{\forall i,j\in S|i=n} w_{i,j}^{r}=0 \qquad \forall r\in D \quad \forall n\in N\big|n=T_r \tag{8.5}$$

$$\sum_{\forall i,j\in S|j=n} w_{i,j}^{r}-\sum_{\forall i,j\in S|i=n} w_{i,j}^{r}=0 \qquad \forall r\in D \quad \forall n\in N\big|n\notin\{O_r,T_r\} \tag{8.6}$$

$$w_{i,j}=\sum_{\forall r\in D} w_{i,j}^{r} \qquad \forall i,j\in S \tag{8.7}$$

$$w_{i,j}\le K\cdot\delta_{i,j} \qquad \forall i,j\in S \tag{8.8}$$

$$\delta_{i,j}=\delta_{j,i} \qquad \forall i,j\in S \tag{8.9}$$

$$\delta_{i,j}\le 1 \qquad \forall i,j\in S \tag{8.10}$$

The constraints in equations (8.2) through (8.6) are the standard flow-balance constraints of the transhipment problem. Equations (8.2) and (8.3) assert that for each demand relation *r*, the total source flow from the origin of the demand is the total demand, and the total flow into the origin of the demand is zero, respectively. Equations (8.4) and (8.5) assert the opposite: the total sink flow into the destination node of a demand is the total demand and the total flow out of the destination is zero. Equation (8.6) ensures that there is no net sourcing or sinking of flow at nodes that are not the origin and destination of the demand (i.e., transhipment nodes). The need to express the concept of transhipment at these nodes (net incoming flow equals net outgoing flow for a given commodity) is ultimately why the whole formulation, including capacities, flows, and edge selection variables, is forced into a directional framework.

The constraints in equation (8.7) ensure that enough working capacity is assigned to each edge to accommodate all of the flows simultaneously passing over it. Since demands are represented to be directional, so too are the spans in the network and consequently, working capacity is assigned to directional edges for use by flow on the span in that direction only. In the context of a transport network where demands and spans are considered to be bi-directional, the working capacity on a span between nodes *i* and *j* (no direction implied) is simply the sum of the working capacities in both directions $(=w_{i,j}+w_{j,i})$. In some versions of the problem where explicitly knowing the edge capacities that result isn't important, this constraint set can be eliminated and referred to the objective function by including an additional summation over all demand relations, so that the objective function becomes $\sum_{\forall i,j\in S}\left(F_{i,j}\cdot\delta_{i,j}+c_{i,j}\cdot\sum_{\forall r\in D}w^{r}_{i,j}\right)$.

As described above, the set *S* represents the set of *eligible* spans that could be used in the network topology. In equation (8.8), an edge decision variable is forced to be $\delta_{i,j}=1$ as long as working capacity is assigned to it, or $w_{i,j}>0$ (it can't be larger than 1 because of equation (8.10)). An edge *i,j* is selected into the topology if $\delta_{i,j}=1$, in which case the fixed charge, $F_{i,j}$, for the associated edge contributes to the objective function in equation (8.1). Equation (8.9) ensures that if a directional edge exists from node *i* to node *j*, then the equivalent edge in the opposite direction is also asserted to exist. Since this will effectively cause the objective function to charge $F_{i,j}$ and $F_{j,i}$, we can set either $F_{i,j}=0$ or $F_{j,i}=0$, or alternatively we can set both $F_{i,j}$ and $F_{j,i}$ to equal half of what the true fixed charge is for the bi-directional edge they represent. It is the interaction of the constraints in equations (8.9) and (8.10) that effectively map the directional structure inherent in the problem into a bi-directional transport network form.

In our version of the FCR problem, we model a single fixed charge, $F_{i,j}$, associated with acquiring the right of way on which the fibre facility route is established, installing the conduit and fibre cables, and all other one-time costs that might be incurred. Any number of integral capacity additions on the edge, representing the establishment of each new DWDM transmission system, is each charged an equivalent amount, $c_{i,j}$, no matter whether it is the first to be turned on or the 100th. We use an *edge-to-unit-capacity cost ratio* parameter, Ω, to represent the ratio of the fixed charge of establishing an edge to the per-unit cost of adding working capacity to that edge ($\Omega = F_{i,j}/c_{i,j}$). In some of the tests that follow, we vary Ω to investigate its effect on our designs. In practice, capacity on an edge may also have a secondary growth structure in steps representing costs associated with equipping individual new channels on a DWDM transmission system. For present purposes we avoid this extra dimensionality since the approximation is minor in terms of the basic effects involved. A single capacity step can be interpreted as representing either a per-channel average step cost that includes pro-rating the larger per-system cost step, or conversely that each integral step corresponds to a system addition at an assumed average fill level of per-channel steps, or simply that the entire system is fully channel-equipped when placed.

Other versions of the problem may involve a family of capacity units or modularity-type constraints [24], without there being a single dominant get-started edge cost and smaller capacity unit steps. The topology design problem in [38] uses this method, although their model is in the form of an arc-path ILP. We again note that our network flow treatment of the problem avoids any need to explicitly enumerate eligible working routes as would be needed in an arc-path representation of the problem. In a situation where the set of eligible spans is quite large (it could potentially include up to all $N \cdot (N-1)/2$ possible bi-directional edges in the graph), such an enumeration is virtually impossible for all but the very smallest networks of a few nodes.

The FCR problem may also be further generalized to include pre-existing edges. It is easy to add such specific considerations by representing existing edges as having zero edge cost ($F_{i,j} = F_{j,i} = 0$), or with an added equality constraint that directly asserts the respective edge decision variable in the solution ($\delta_{i,j} = \delta_{j,i} = 1$). In some cases, there may even be some available capacity, $w'_{i,j}$, already installed on a span. We can represent this by adding an extra constraint forcing the working capacity on that span to be at least that amount ($w_{i,j} + w_{j,i} \geq w'_{i,j}$) and then discounting the objective function by $w'_{i,j} \cdot c_{i,j}$ since that capacity will not cost anything. This may also require us to change the strict equalities in equation (8.7) to inequalities ($w_{i,j} \geq \sum_{\forall r \in D} w'_{i,j}$)

in case the optimal design would need less than the available capacity on any particular span(s).

The un-capacitated version of the FCR problem is addressed in [19], with an emphasis on solving it to optimality through a new criterion for use in the branch-and-bound search. The version of FCR that becomes a constituent part of our problem, however, is capacitated, though not in the sense that we will assert pre-existing capacities or limits, but rather in the aspect that edge capacities will be integral. As a consequence there are mutual capacity constraints (equation (8.6)) governing the composite routing solution under the discrete capacity on each edge in the design. It is pointed out in [39] that it is these mutual capacity constraints that make the capacitated versions of FCR "NP-hard and very difficult to solve in practice". A survey of other formulations and solution approaches for the capacitated multi-commodity FCR problem is also provided in [39], and numerous forms of ILP relaxations and approximations are also given. The solution gaps vary somewhat unpredictably over the five relaxation strategies tested, and reach to as high as 40%. They are rarely better than a Tabu Search heuristic for the same problems, which affirms the computational difficulty of capacitated multi-commodity FCR problems and even of getting good bounds for the problem.

One of the difficulties in applying branch-and-bound to solve FCR problems is that a strong relaxation (dropping all integrality constraints, including on the edge variables) gives very weak lower bounds, because the mutual capacity constraints are so crucial to determining an optimal FCR solution. In the un-relaxed FCR problem, the choice of routes for each working flow is strongly coordinated with that of other flows, so as to use as few edges and capacity units as is optimal. We will later see that this is abundantly true of the complete topology design problem with restoration as well (see Section 8.2), because it inherits this aspect of FCR and adds to it similar aspects of sharing spare capacity for restoration, which are intimately dependent on the graph topology. Under the relaxation each flow is more or less independently routed since there is no shared-efficiency effect from the fixed charge component. In other words, the solution space of an FCR problem is strongly and discretely structured by the topology variables. If we completely relax the edge decision variables, then a form of amorphous uncoupled sea of flows is represented with total costs that are almost completely unrelated to the real problem on a discrete graph. This is why relaxation of the 1/0 edge decision variables gives an almost meaningless and extremely loose lower bound.

The work in [39] also looks at adding a constraint to the FCR problem that forces the solution to contain at least N-1 edges, which is simply required to have a connected network. That minimum connectivity can be asserted by addition of the optional constraint in equation (8.11).

$$\sum_{\forall i,j \in S} \frac{\delta_{i,j}}{2} \geq |N| - 1 \tag{8.11}$$

Optionally, we can also incorporate an a priori expectation that in practice, cost-optimal solutions lie with solution graphs of limited maximum nodal degree. In other words, we know that in real networks, there is some upper level of connectivity, $\bar{d}_{max}$, that is not plausible. This optional constraint can be implemented by the addition of equation (8.12).

$$\sum_{\forall i,j \in S} \frac{\delta_{i,j}}{2} \leq \bar{d}_{max} \tag{8.12}$$

In summary, there is a considerable body of literature, methods and software available to solve FCR problems. This is desirable and relevant to the present work because the heuristic solution method to follow effectively reduces the full problem of topology, routing and survivability to a special instance of classical FCR plus two other new, but easier to solve sub-problems.

8.2 MESH TOPOLOGY, ROUTING, AND SPARING

In much the same way that we expanded on the SCA formulations to form JCA formulations in Section 5.1, we now expand on the FCR formulation provided in Section 8.1.1 to provide for restoration routing and spare capacity allocation in addition to the topology and working routing already implemented. We call this new problem the *mesh topology, routing, and sparing* (MTRS) problem. The MTRS model is specifically based on span restoration as described in Section 4.4.1, and we include all single span failures as the set of failure scenarios.

Again, the general MTRS model assumes that all possible $N \cdot (N-1)/2$ bi-directional edges (i.e., all $N \cdot (N-1)$ directional edges) are candidates for selection in the final network topology. We must therefore continue to use a network flow model, rather than an arc-path model, which is a significant departure from earlier work on restorable network design. When the topology has been defined ahead of time, as is the case in prior methods, an arc-path approach is usually preferred because it allows explicit control and direct inspection of the working and restoration routes employed in the solution. It also allows a trade-off between solution quality and run times through strategies that control or ration the total number of eligible routes represented for working and restoration flow assignment in such problems (see Section 5.2).

However, when the graph topology itself is admitted as a decision variable, the preprocessing required for an arc-path formulation becomes untenable because a master set of eligible routes would have to be developed for representation that is structured in some way so that, for each combination of edges selected, it is evident which routes on the full-mesh graph, are enabled under the specific set of non-zero edge variables. It is as though every plausible topology would have to be identified ahead of time and a set of eligible working and restoration routes determined and stored for each topology instance. Hence we are virtually forced to use network flow representation of the working path routing and restoration flow solutions because of its self-contained nature.

8.2.1 MTRS ILP FORMULATION

To express the complete MTRS ILP formulation, we use the sets, input parameters, and decision variables that have already been defined for the FCR formulation, and we add the following.

New Decision Variables:

- $s_{i,j}^{k,l} \geq 0$ is the amount of restoration flow routed over the edge between nodes *k* and *l* (in the direction from *k* to *l*) for restoration of the failed edge between nodes *i* and *j*.

- $s_{i,j} \geq 0$ is the spare capacity assigned to the directional edge between nodes *i* and *j* to support the largest combination of simultaneously imposed restoration flow requirements over that edge in the *i* to *j* direction.

The formulation itself includes most of the constraint systems used by the FCR formulation, but rather than referring the reader to that earlier model, we repeat them here so as to be able provide the complete model here. The complete MTRS formulation is expressed as follows.

$$\text{Minimize} \quad \sum_{\forall i,j \in S} \left(c_{i,j} \cdot \left(w_{i,j} + s_{i,j} \right) + F_{i,j} \cdot \delta_{i,j} \right) \tag{8.13}$$

$$\text{Subject to:} \quad \sum_{\forall i,j \in S | i=n} w_{i,j}^{r} = d^{r} \qquad \forall r \in D \quad \forall n \in N | n = O_r \tag{8.14}$$

$$\sum_{\forall i,j \in S | j=n} w_{i,j}^{r} = 0 \qquad \forall r \in D \quad \forall n \in N | n = O_r \tag{8.15}$$

$$\sum_{\forall i,j \in S | j=n} w_{i,j}^{r} = d^{r} \qquad \forall r \in D \quad \forall n \in N | n = T_r \tag{8.16}$$

$$\sum_{\forall i,j \in S | i=n} w_{i,j}^{r} = 0 \qquad \forall r \in D \quad \forall n \in N | n = T_r \tag{8.17}$$

$$\sum_{\forall i,j \in S | j=n} w_{i,j}^{r} - \sum_{\forall i,j \in S | i=n} w_{i,j}^{r} = 0 \qquad \forall r \in D \quad \forall n \in N | n \notin \{O_r, T_r\} \tag{8.18}$$

$$w_{i,j} = \sum_{\forall r \in D} w_{i,j}^{r} \qquad \forall i,j \in S \tag{8.19}$$

$$\sum_{\forall i,k \in S | j \neq k} s_{i,j}^{i,k} = w_{i,j} \qquad \forall i,j \in S \tag{8.20}$$

$$\sum_{\forall k,j \in S} s_{i,j}^{k,i} = 0 \qquad \forall i,j \in S \tag{8.21}$$

$$\sum_{\forall k,j \in S | i \neq k} s_{i,j}^{k,j} = w_{i,j} \qquad \forall i,j \in S \tag{8.22}$$

$$\sum_{\forall j,k \in S} s_{i,j}^{j,k} = 0 \qquad \forall i,j \in S \tag{8.23}$$

$$\sum_{\forall n,k \in S | k \notin \{i,j\}} s_{i,j}^{n,k} - \sum_{\forall k,n \in S | k \notin \{i,j\}} s_{i,j}^{k,n} = 0 \qquad \forall i,j \in S \quad \forall n \in N | n \notin \{i,j\} \tag{8.24}$$

$$s_{k,l} \geq s_{i,j}^{k,l} \qquad \forall i,j \in S \quad \forall k,l \in S | k,l \neq i,j \tag{8.25}$$

$$s_{k,l} \geq s_{j,i}^{k,l} \qquad \forall i,j \in S \quad \forall k,l \in S | k,l \neq i,j \tag{8.26}$$

$$w_{i,j} + s_{i,j} \le K \cdot \delta_{i,j} \qquad \forall i, j \in S \qquad (8.27)$$

$$\delta_{i,j} = \delta_{j,i} \qquad \forall i, j \in S \qquad (8.28)$$

$$\delta_{i,j} \le 1 \qquad \forall i, j \in S \qquad (8.29)$$

The constraints in equations (8.14) through (8.19) are the same constraints as those in equations (8.2) through (8.7) of the FCR formulation, and serve the same purpose; they are the simultaneous multi-commodity flow-balance and capacity-allocation constraints of the capacitated transhipment problem that deal with the routing of working flows. For each demand relation, there is a pair of source node constraints in equations (8.14) and (8.15), asserting sufficient net sourcing of flow and no net sinking of flow at the source node, respectively, as well as a pair of sink node constraints in equations (8.16) and (8.17), asserting sufficient net sinking of flow and no net sourcing at the sink node, respectively. The constraints in equation (8.18) apply the transhipment nature of intermediate nodes that are neither a source or sink for the particular demand relation, and equation (8.19) generates the working capacity allocation on each edge so as to simultaneously support the required working flow variables on each edge, for each demand relation.

The transportation-like network flow problem structure is also evident in equations (8.20) through (8.24). This is a set of *non*-simultaneous *single*-commodity network flow sub-problems, each describing the corresponding source, sink and transshipment constraints pertaining to the restoration flows for one particular edge failure. Equations (8.25) and (8.26) are the corresponding spare capacity generating constraints. Like the equivalent constraint sets in standalone SCA, equations (8.25) and (8.26) use inequalities because the requirement is to force the spare capacity on each edge to satisfy the largest of the non-simultaneous restoration flows imposed on the given edge. Equation (8.27) replaces equation (8.8) from the FCR formulation to deal with the edge selection variables that define the topology on which the above routing and restoration solutions are jointly coordinated to minimize total cost. Equations (8.28) and (8.29) are carried over from equations (8.9) and (8.10) in the FCR formulation.

If we wish, we can also include the optional constraints in equations (8.11) and (8.12) to assert selection of enough edges to close the graph, and to disallow excessively connected topologies, respectively. In the context of a fully restorable network required here, we can extend the principle in equation (8.11) to assert advance knowledge that any feasible graph must not only be connected, but also closed, so there must be at least *N* edges. The corresponding solution with exactly *N* edges is a Hamiltonian ring, which interestingly, does actually emerge in MTRS test cases where fixed charges are significantly higher that the incremental routing costs (and where a Hamiltonian cycle exists). We can go even one step further by adding a constraint requiring each node to be individually of at least degree-2, so that failure of any one edge cannot disconnect any node. These optional constraints can be implemented by the addition of equations (8.30) and (8.31), respectively, to the MTRS formulation.

$$\sum_{\forall i,j \in S} \frac{\delta_{i,j}}{2} \geq |N| \qquad (8.30)$$

$$\sum_{\forall i,j \in S | i=n} \delta_{i,j} \geq 2 \qquad \forall n \in N \qquad (8.31)$$

The addition of the constraints in equations (8.30) and (8.31) (and (8.12)) are not logically required parts of the MTRS problem, but can speed up the branch and bound solution times by expressing topological properties that have to exist in any connected network that satisfies the restorability constraints in equations (8.20) through (8.24).

Whereas equations (8.30) and (8.31) may or may not be applied, they are certainly mathematical truths. On the other hand, equation (8.12) is a "belief-based" optional constraint, which represents the a priori knowledge that no known transport network has an average nodal degree higher than some $\bar{d}_{max}$. In other words, if we put credence in the merit of real transport network graphs for their intended purposes, we can derive a guideline on the maximum number of edges an optimal design could plausibly contain. In practice we do believe that with current technologies and costs, optimal graphs lie somewhere in the range $2 \leq \bar{d} \leq 5$ (so $\bar{d}_{max} = 5$), which is where nearly all published examples of transport networks exist. Of course in a purely general instance of MTRS as a mathematical problem only, it could not be known a priori what value of $\bar{d}_{max}$ brackets the optimum and so use of equation (8.12) might not be advisable. But in problems where the costs of edges and capacities are derived from real circumstances, it may be quite reasonable and useful and to apply equation (8.12) with perhaps a $\bar{d}_{max} = 6$ (or certainly $\bar{d}_{max} = 8$) to restrict the solution space without affecting optimality.

The binary edge-selection variables are fundamental to the mutual routing and capacity allocation in a real network, and allowing them to take on values other than $\delta_{i,j} = 1$ or $\delta_{i,j} = 0$ will have no real meaning, so their integrality cannot be relaxed. In our solutions, integrality of working and spare capacity variables is also upheld to correspond to integer wavelength channels. Underlying working and restoration flow variables, on the other hand, are relaxed and allowed to take on real (i.e., non-integer) values. As already discussed in Section 5.1.1, as long as integrality is asserted on the spare capacity variables, the integrality requirement on the restoration flow variables can be relaxed without affecting solution quality or feasibility [3], [117]. In this case each restoration flow sub-problem for an individual failure scenario is a single-commodity integer-capacitated network flow problem for which flows remain integral if demands and capacity are integral. The relaxation of working flows is justified as an acceptable practical measure when attempting direct solution of the complete MTRS problem. Fractional working flows may arise in the solutions but our own experience, as well as work in [70], indicate that a simple repair procedure can re-integrate fractional working flows at minimal or no impact on the objective function cost. When faced with the same issue, the authors of the work in [93] indicate that the gap due to working flow relaxations is only approximately 1% in their experience. If any relaxations are to be considered at all, the choice of integer working and spare capacities with relaxed flows is advantageous over integrality on flows with relaxed capacities. This is because there is only one capacity variable per edge, but there is one working flow variable on each edge for each demand pair in the problem and a restoration flow variable on each edge for each other edge in the graph.

8.2.2 INITIAL MTRS TESTING

Initial testing of the MTRS network design model was conducted on the six master network topologies described in Section 6.1, with each span in those topologies representing a candidate edge in the MTRS problem. For further testing, each master network was also transformed into a full-mesh version where node positions remain the same but every possible node-pair is connected with a candidate edge with incremental capacity costs equivalent to the Euclidean distances between their end nodes (just as the spans in the original master networks themselves). We name these new networks 15n30s1-Full, 20n40s1-Full, 25n50s1-Full, 30n60s1-Full, 35n70s1-Full, and 40n80s1-Full, and applied the same set of uniform random working demands (from 1 to 10 lightpaths per O-D pair) as those we applied to the basis master networks. A group of tests were carried out for each network topology based on a variety of different edge-to-unit-capacity cost ratios, $\Omega = F_{i,j}/c_{i,j}$.

8.2.2.1 EXPERIMENTAL STUDY METHOD

As with the benchmarking designs in CHAPTER 6, the MTRS network design model was implemented in AMPL Mathematical Programming Language version 9.0 and solved with CPLEX 9.0 MIP Solver running on a 4-processor SUN UltraSparc III running at 900 MHz with 16 GB of RAM. Pre-processing for eligible working and restoration routes was not required, but data files containing network topology information and other input parameters were prepared on a dual-processor AMD Opteron 242 PC with 1 GB of RAM, running Windows 2000. All working and spare capacity allocations were integer, corresponding to capacity design and restoration mechanisms at the wavelength level, and edge decision variables $\delta_{i,j}$ were strictly binary. Because of the complexity of the MTRS ILP model, most results are based on time-limited CPLEX runs with various MIPGAP settings ranging from 10^{-2} (i.e., solutions are guaranteed to be within 1% of optimal) to 0.20 (i.e., a fully terminated solution would have an optimality gap of 20%). The recorded runtimes are actual elapsed time, which is roughly equivalent to actual CPU time on the four-processor unit as a whole, where all four processors are devoted to the CPLEX task full time. Actual runtimes and optimality gaps are provided in the appropriate tables that follow.

8.2.2.2 CHOOSING SUITABLE EDGE-TO-UNIT CAPACITY COST RATIOS

In Section 8.1.1, we introduced the edge-to-unit-capacity cost ratio parameter, $\Omega = F_{i,j}/c_{i,j}$. In reality, Ω can vary widely from network to network, and possibly even from span to span within a network, and would be dependent in part on spans' fixed costs for rights-of-way and lease acquisitions, excavation, duct installation, and so on, relative to incremental per-channel costs for fibre, terminations, etc., and as well as on the choice of transmission systems used and definitions of unit-capacity sizes. In our design model, the exact value for Ω could even depend on less tangible factors such as the network planner's own experience, political considerations, and/or biases towards building networks of high or low average nodal degree.

Based on indications from industry colleagues, values in the order of $\Omega \approx 10$ to $\Omega \approx 1000$ are not out of the question when considering wavelength unit capacities in an optical network, and our experiments will select Ω values in that range. Various values for Ω were investigated for our test case networks, and to aid in the illustration of our design methods, we selected three Ω values for each network family such that when we calculate the total (capacity plus topology) design costs of all 163 test case networks, we produce total cost curves with reasonably well-defined minimums. Those Ω values are shown in Table 8.1.

Table 8.1 – Edge-to-unit-capacity cost ratios for test case network families.

15n30s1	20n40s1	25n50s1	30n60s1	35n70s1	40n80s1
100	250	250	250	500	500
150	500	500	500	750	750
200	750	750	750	1000	1000

For reference purposes, we also calculate the total network design cost for each test network when it is optimally designed using the span-restorable JCA model (i.e., the benchmark designs obtained in CHAPTER 6) and with span fixed costs corresponding to the various Ω values from Table 8.1. Each data point in Figure 8.2 through Figure 8.7 represents the total network design cost of the network of the indicated family and nodal degree with the specified Ω value. In each figure, the data points have been normalized to the cost of the lowest cost member of the network family with the smallest Ω value. As expected, network costs tend to be higher on either end at high and low average nodal degree, and lower in networks with intermediate average nodal degrees. While it is more obvious in some networks than in others, and many curves have several local minima, the same general trend exists in all curves. "Hollow" markers indicate the lowest cost network within each curve.

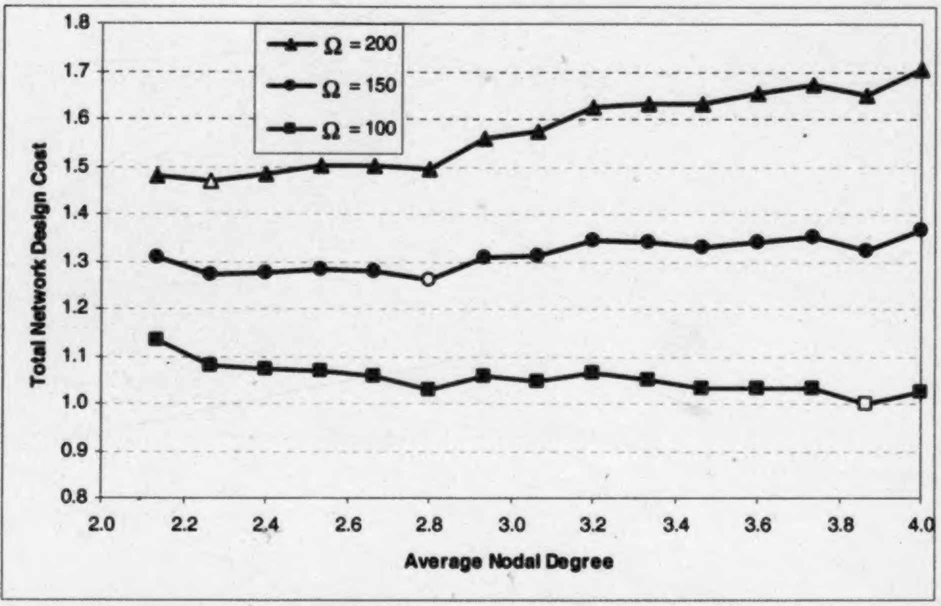


Figure 8.2 – Complete network design costs for the 15n30s1 network family with various edge-to-unit-capacity cost ratios.

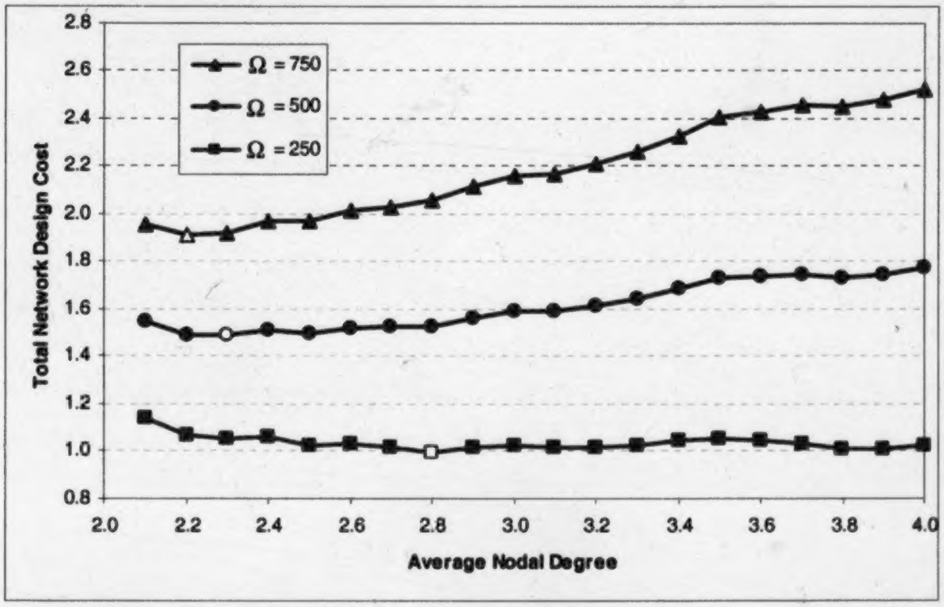


Figure 8.3 – Complete network design costs for the 20n40s1 network family with various edge-to-unit-capacity cost ratios.

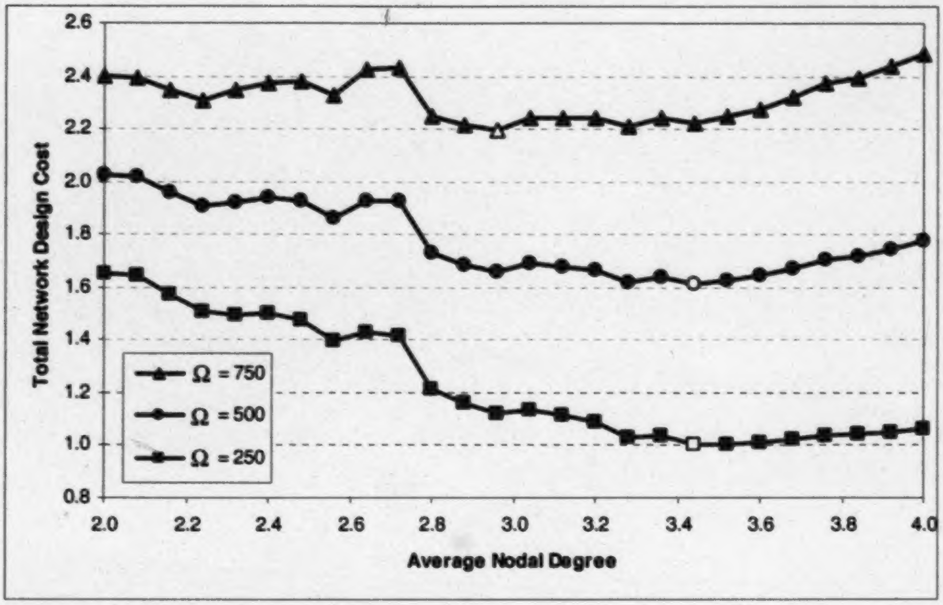


Figure 8.4 – Complete network design costs for the 25n50s1 network family with various edge-to-unit-capacity cost ratios.

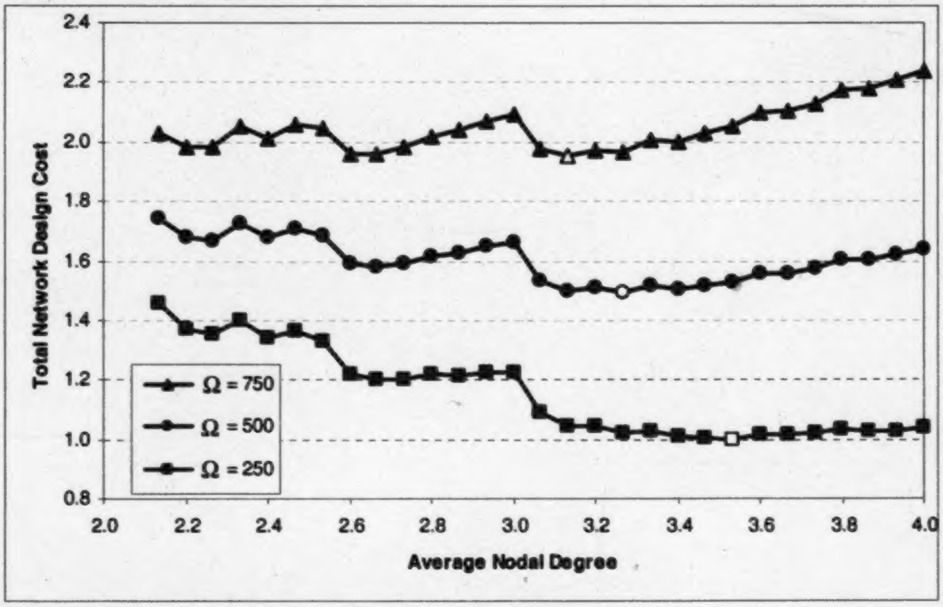


Figure 8.5 – Complete network design costs for the 30n60s1 network family with various edge-to-unit-capacity cost ratios.

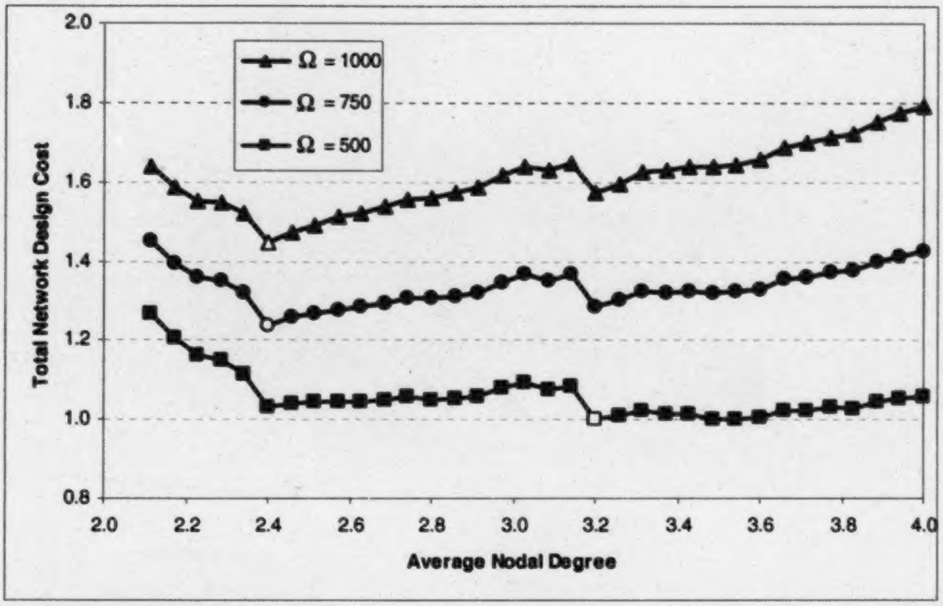


Figure 8.6 – Complete network design costs for the 35n70s1 network family with various edge-to-unit-capacity cost ratios.

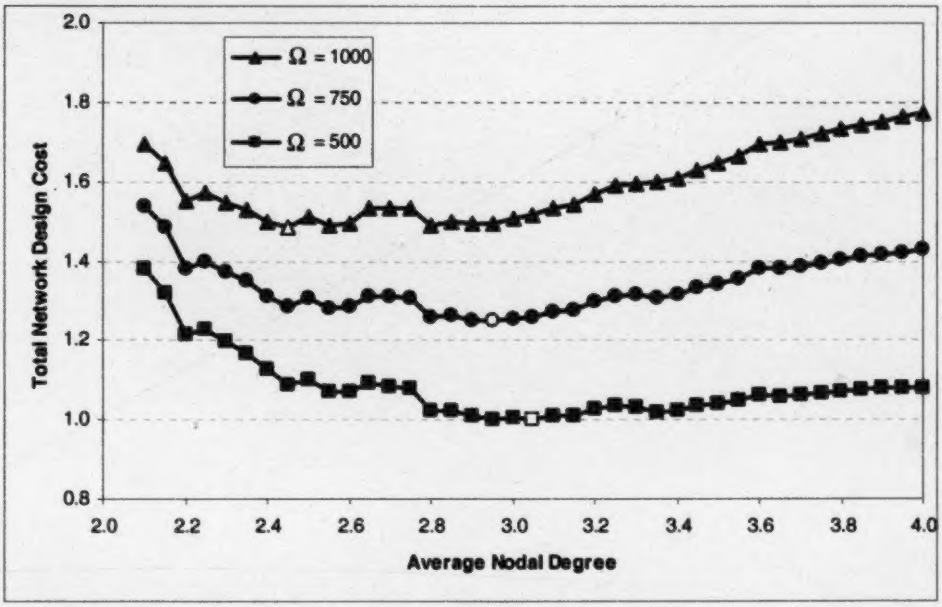


Figure 8.7 – Complete network design costs for the 40n80s1 network family with various edge-to-unit-capacity cost ratios.

8.2.2.3 RESULTS AND DISCUSSION

A summary of the MTRS solutions of the six master networks is shown in Table 8.2. Each master network was solved with the three Ω values from Table 8.1, so in total there were 18 test cases. For each test case, the benchmark reference solution is equivalent to the minimum cost network of the corresponding family with the indicated Ω value as determined in the figures in Section 8.2.2.2 (i.e., the "hollow" marker data points in Figure 8.2 through Figure 8.7). The "Benchmark" "# Spans" and "Cost" columns represent the number of spans and the normalized total cost of the lowest cost network of the appropriate network family. The next three columns detail the number of spans in the resulting MTRS solution ("# Span"), its average nodal degree ("$\bar{d}$"), and its total network design cost ("MTRS Sol'n"), the latter of which is normalized to the same scale as the reference solution. The "% Change" column is the amount by which the MTRS solution improves on the reference solution. The runtimes given are only the CPLEX runtimes, and do not include any preprocessing of data files or AMPL processing prior to handing the problem to CPLEX. The "MIP Gap" column is the gap to optimality of the MTRS solution, and is equal to the provably maximum difference between the solution obtained and the true optimal solution *as a percentage of the obtained solution*. So for instance, the first test case has a MIP gap of 1% meaning the strictly optimal solution is only 1% lower than the MTRS solution given. Because the complexity of the problem scales exponentially with the size of the network, the larger test cases could not be as easily solved within small MIP gaps as the smaller test cases, so MIP gaps generally increase with the size of the master network. The intent here, however, is not necessarily to obtain strictly optimal MTRS solutions, but rather to demonstrate that such solutions are generally quite superior to the reference solutions based on the individual members of the test case network families, and also to show that obtaining such solutions is not easy.

For the four smallest master networks, the MTRS solutions are an average of 18.5% less costly than the best reference solutions from the individual test networks of each family, and in the best case was 27.7% better. We can note, however, that the reference solutions are the benchmark span-restorable JCA designs obtained in CHAPTER 6, which were solved using the arc-path model of Section 5.1.1 with pre-selection of five eligible working routes per demand relation and ten eligible restoration routes per span. The MTRS solutions, on the other hand, are based on the transhipment ILP model from Section 8.2.1, which doesn't explicitly enumerate eligible working and restoration routes but rather implicitly considers *all* such routes that can possibly be drawn within the network graph. So at least a small amount of that improvement can be due to the improved working and restoration routing provided by a transhipment model, but tests show that these differences only account for approximately 1.66% of that reduction in total cost[15]. Most of the 18.5% average reduction in total cost is therefore due to the topological optimization provided by MTRS.

Table 8.2 – MTRS solutions of all six master networks, each with three different edge-to-unit-capacity cost ratios.

			Benchmark		MTRS Solutions					
#	Test Network	Ω	# Spans	Cost	# Spans	$\bar{d}$	MTRS Sol'n	% Change	Runtime (hours)	MIP Gap
1	15n30s1	100	29	1.00	21	2.80	0.83	16.6%	0.40	1%
2	15n30s1	150	21	1.26	19	2.53	1.02	18.9%	0.55	1%
3	15n30s1	200	17	1.47	19	2.53	1.20	18.1%	0.50	1%
4	20n40s1	250	28	1.00	26	2.60	0.76	23.6%	8.3	5%
5	20n40s1	500	23	1.49	25	2.50	1.16	22.1%	8.7	5%
6	20n40s1	750	22	1.91	23	2.30	1.55	18.9%	8.6	5%
7	25n50s1	250	43	1.00	36	2.88	0.85	15.2%	3.4	10%
8	25n50s1	500	43	1.61	32	2.56	1.26	21.5%	6.0	13.6%
9	25n50s1	750	37	2.20	31	2.48	1.59	27.7%	6.0	10.3%
10	30n60s1	250	53	1.00	46	3.07	0.92	8.3%	7.7	20%
11	30n60s1	500	49	1.49	42	2.80	1.33	11.1%	5.9	20%
12	30n60s1	750	47	1.95	38	2.53	1.56	20.3%	6.0	20%
13	35n70s1	500	56	1.00	64	3.66	1.03	-2.9%	12.0	28.9%
14	35n70s1	750	42	1.24	68	3.89	1.43	-15.2%	12.0	37.8%
15	35n70s1	1000	42	1.44	66	3.77	1.72	-19.3%	12.0	38.8%
16	40n80s1	500	61	1.00	78	3.90	1.08	-8.0%	12.0	31.4%
17	40n80s1	750	59	1.25	78	3.90	1.41	-12.7%	12.0	36.8%
18	40n80s1	1000	49	1.49	79	3.95	1.78	-20.0%	12.0	41.3%

Although the MTRS solutions do show significant improvements over the benchmark reference solutions (for the four smallest networks), they also took a considerably long time to solve, even for small test case networks and with large optimality gaps. The 15-node test cases did solve quite quickly (approximately 30 minutes for each) to within a gap to optimality of only 1%, but the 20-node master network test cases for instance, took over 8 hours each with MIP gaps of 5%. The 30-node master network needed approximately 6 hours or more just to solve within a 20% gap to optimality. Perhaps the most telling data in Table 8.2 is that of the two largest master networks (35n70s1 and 40n80s1). Even given a 12 hour runtime, the MTRS problem could only be solved to within 28.9% of optimal in the best case for these larger networks, and in all cases, the MTRS solution was actually worse than the best reference solution obtained from the individual members of each test case network family (13.0% worse on average).

---

[15] A transhipment ILP model was used to produce span-restorable JCA network designs with optimal working and spare capacity placement in each of the 163 test case networks from CHAPTER 6. Equivalent total network design costs were calculated with the various Ω values from Table 8.1, and they were an average of only 1.71% less costly than the reference solutions shown in Figure 8.2 through Figure 8.7. For the four smallest master networks, the difference was slightly smaller, at 1.66%.

Solutions of the *full-mesh* versions of the six master networks were also attempted, but in 12 hours of runtime, solutions were available for only the 15n30s1-Full test case network. The data in Table 8.3 lists the name of the master network converted to a full mesh ("Test Network"), the Ω value used, the benchmark reference solution ("Ref.Sol'n"), the previous MTRS solution of the non-full-mesh master network ("Prev.Sol'n"), and details of the full-mesh MTRS solution including the number of spans in the design solution, the average nodal degree, the total cost of the solution, and the percentage change relative to the "Prev. Sol'n" value, as well as the runtime and optimality gap. In the "% Change" column, the negative values indicate that the full-mesh MTRS solutions were actually worse than the non-full-mesh solutions.

As shown in Table 8.3, the full-mesh MTRS solutions for the 15n30s1-Full network had 20.8% optimal gaps at best after 12 hours, and were actually 7.3% worse than the equivalent MTRS solutions of the 15n30s1 master network itself. None of the larger full-mesh test cases even yielded feasible solutions in 12 hours of runtime. For the 20n40s1-Full test case network, 24 hours of runtime provided solutions that were an average of 434.6% worse than the previous best solution, with optimality gaps of 85.4%. For the 20n40s1-Full network with $\Omega = 250$, for instance, the MTRS solution was nearly four times the cost of the best previous solution (i.e., a 275.6% increase over the MTRS solution of the 20n40s1 master network) and had an 81.8% gap to optimality. An optimality gap that large means that the solution is essentially worth-less since the *true* optimal design could actually have a cost anywhere between 2.87 (the normalized cost of the time-limited solution itself) and 0.52 (or 81.8% below 2.87). 24 hours of runtime did not even produce a feasible solution at all for the 25-node or larger full-mesh topologies. Clearly, the solutions of the complete MTRS problem are very difficult to obtain in any reasonable amount of time for all but the smallest of networks, and the use of heuristic methods would certainly seem warranted.

Table 8.3 – MTRS solutions of 15n30s1-Full and 20n40s1-Full full-mesh networks, each with three different edge-to-unit-capacity cost ratios.

					Full-Mesh MTRS Solutions					
#	Test Network	Ω	Ref. Sol'n	Prev. Sol'n	# Spans	$\bar{d}$	Opt. Sol'n	% Change	Runtime (hours)	MIP Gap
1	15n30s1	100	1.00	0.83	24	3.20	0.87	-4.5%	12	20.8%
2	15n30s1	150	1.26	1.02	22	2.93	1.10	-7.5%	12	23.8%
3	15n30s1	200	1.47	1.20	20	2.67	1.32	-10.0%	12	24.5%
4	20n40s1	250	1.00	0.76	105	10.50	2.87	-275.6%	24	81.8%
5	20n40s1	500	1.49	1.16	105	10.50	5.49	-373.0%	24	84.4%
6	20n40s1	750	1.91	1.55	134	13.40	11.70	-655.1%	24	90.1%

Although the data in Table 8.2 includes average nodal degree and normalized costs, it is not easy to visualize where the MTRS solutions fit in with the reference solutions of Figure 8.2 through Figure 8.7. In an effort to better understand their relationship with those benchmark solutions, we produce Figure 8.8 through Figure 8.11, in which we have superimposed the MTRS solutions onto the data of Figure 8.2 through Figure 8.5. We exclude the 35n70s1 and 40n80s1 plots since the MTRS solutions for those networks are not solved close enough to optimality to show an improvement over the reference solutions. The new figures are identical to Figure 8.2 through Figure 8.5 except that there is one new isolated data point corresponding to each curve (matching the colour and marker shape of the associated curve), with each new data point representing the related MTRS solution. It is interesting to note that not only are the optimal MTRS solutions significantly better than the reference solutions (as we saw from Table 8.2), but that in general, the $\bar{d}$ of the optimal MTRS solutions do not match the $\bar{d}$ of the optimal benchmark solution curves at all. For instance, the optimal reference solution for the 15n30s1 network family with $\Omega = 100$ has $\bar{d} = 3.87$ but the optimal MTRS solution has $\bar{d} = 2.80$.

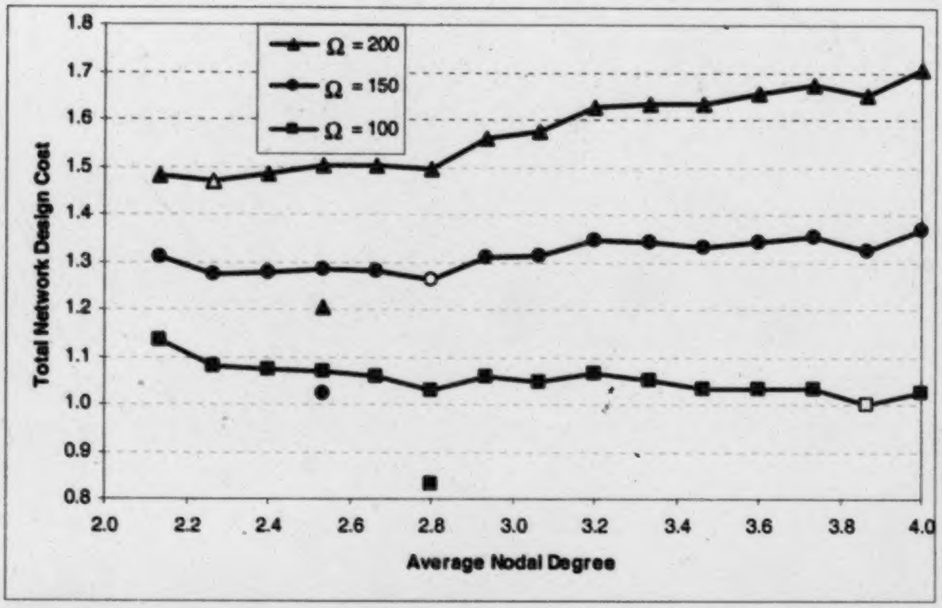


Figure 8.8 – MTRS solutions of the 15n30s1 master network superimposed on the complete network design costs for the entire network family.

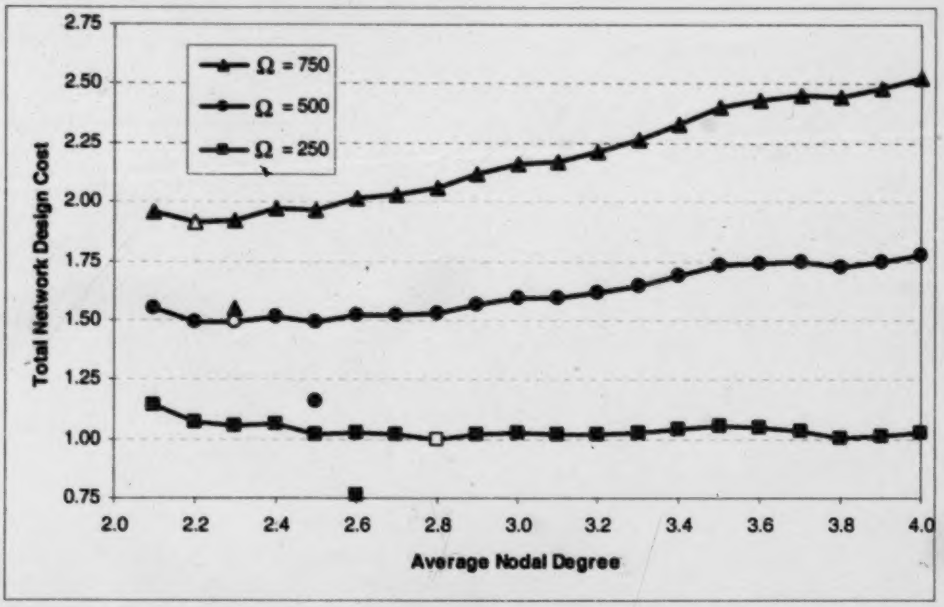


Figure 8.9 – MTRS solutions of the 20n40s1 master network superimposed on the complete network design costs for the entire network family.

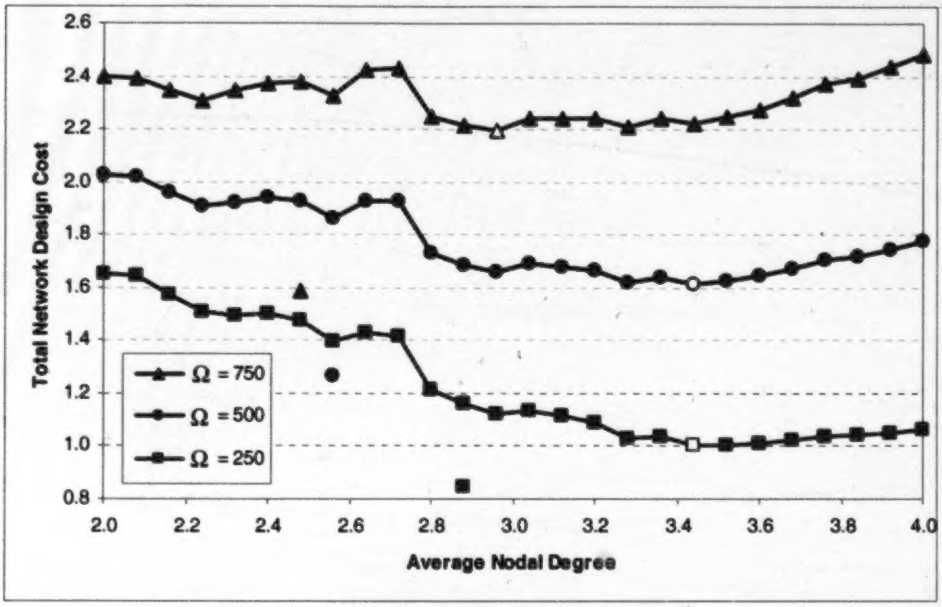


Figure 8.10 – MTRS solutions of the 25n50s1 master network superimposed on the complete network design costs for the entire network family.

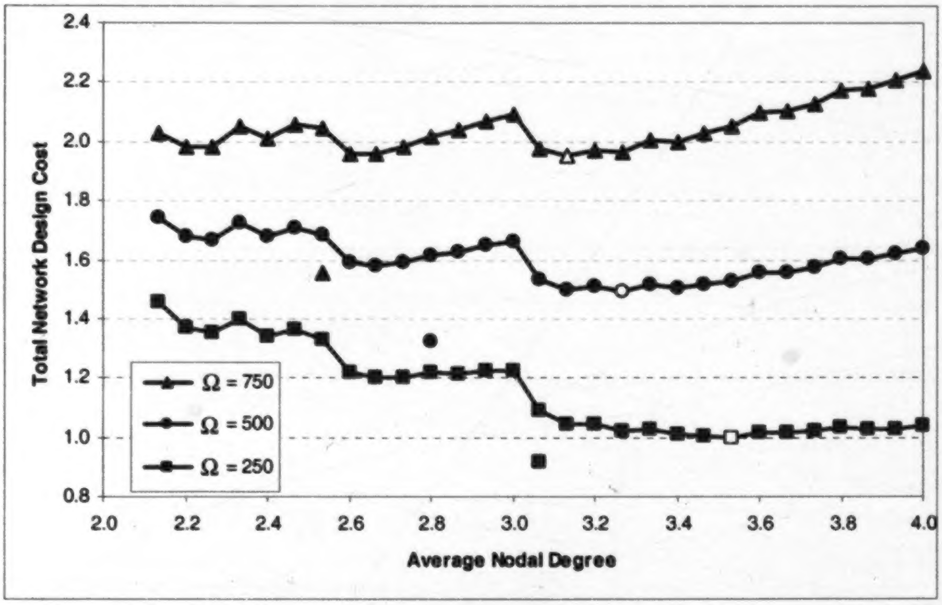


Figure 8.11 – MTRS solutions of the 30n60s1 master network superimposed on the complete network design costs for the entire network family.

The reason is partly because of the manner in which the topologies of the various members of the network families were produced. As described in Section 6.1, starting with the master networks, spans were pseudo-randomly removed one at a time until the topology becomes minimally sparse. As such, each span removal forces the topologies of subsequent members of the family to follow just one path through the space of possible network topologies. The size of that topology space is vast, however. Although some possible networks would prove to be not bi-connected (and hence not restorable) thereby reducing their total count, the 15n30s1 network family could have over 30 million possible unique topologies with $\bar{d} = 2.67$ alone (i.e., 20 spans); there are also similarly large numbers of unique topologies with 21 spans, with 22 spans, and so on. For full-mesh networks, the problem becomes exponentially larger. Work in [89] showed that a 5-node full-mesh network contained 241 unique bi-connected topologies, a 6-node full mesh had 11,468, and a 7-node full mesh had over 1 million. Attempts to fully exhaust the topology space for bi-connected topologies in an 8-node full mesh were unsuccessful and terminated after a 48-hour runtime. On the other hand, the MTRS problem allows the solver to discover whichever specific topology in that entire space provides the lowest cost design. Given the considerable number of feasible topologies possible, it is virtually assured that an essentially random walk through the topology space would discover a topology that closely matched the MTRS optimal solution in both cost and $\bar{d}$.

This is also one of the main reasons (besides the large number of variables and constraints) that MTRS solutions are so difficult to obtain, particularly for full-mesh networks. Before the solver can even perform working and restoration routing and capacity placement, edge-selections must first collectively form a bi-connected graph. However, such a task is not easy since even in the 15-node master network, there are over 1 billion possible combinations of the 30 original spans (2^{30}). While added-value constraints in the MTRS ILP model can be used to restrict the edge-selection to those combinations with 15 or more spans (the minimum number needed for bi-connectivity), that still leaves more than 614 million possible edge-selection combinations $({}^{30}C_{15} + {}^{30}C_{16} + {}^{30}C_{17} + \dots + {}^{30}C_{29} + {}^{30}C_{30})$. Similar calculations show that a 15-node full-mesh network has more than 4×10^{31} edge-selection combinations with 15 or more spans. Only a very small fraction of those form bi-connected graphs, so the solver has to spend a considerable amount of time just looking for a feasible graph, not to mention one that produces a near-optimal design.

In Figure 8.12 through Figure 8.15, we take a closer look at the actual graph topologies of the optimal MTRS solutions and how they compare to the corresponding optimal reference topologies, with each figure illustrating the topologies of a single network family. Within each figure, each of the top three topologies illustrates the lowest cost member of the network family with the indicated Ω value. So for instance, Figure 8.12(a) is the topology of 15n30s1-29s, which is the member of the 15n30s1 family whose total design cost at $\Omega = 100$ is lower than all of the other members of that family. Likewise, Figure 8.12(b) is the 15n30s1-21s topology, which is the best reference solution if $\Omega = 150$, and Figure 8.12(c) is the 15n30s1-17s topology, which is the best reference solution if $\Omega = 200$. The lower three topologies within each figure illustrate the corresponding optimal MTRS solutions topologies, so Figure 8.12(d) is the 21-span topology of the optimal MTRS solution of the 15n30s1 master network at $\Omega = 100$. Likewise, Figure 8.12(e) is the 19-node topology of the optimal MTRS solution with $\Omega = 150$ and Figure 8.12(f) is the 19-node topology of the optimal MTRS solution with $\Omega = 200$.

The first observation we can make is that the optimal or near-optimal MTRS designs tend to be quite a bit sparser than the best reference solutions. In the four test case networks with near-optimal MTRS solutions, 75% of those solutions had fewer spans than the corresponding reference solutions, and on average, an MTRS solution used 13.5% fewer spans than the optimal reference solution (20.6% when only those cases with lower span counts are considered). As discussed previously, the individual network topologies of each family were produced by a pseudo-random walk through the feasible topology space of the family. They were, however, consistent with topologies that a network planner might consider reasonable since the choice about which span is removed at each step was made by a human designer rather than by a purely random process. So because most of the MTRS solutions tend to be sparser than our benchmark topologies, this suggests that a human-designed network topology is likely to include too many spans.

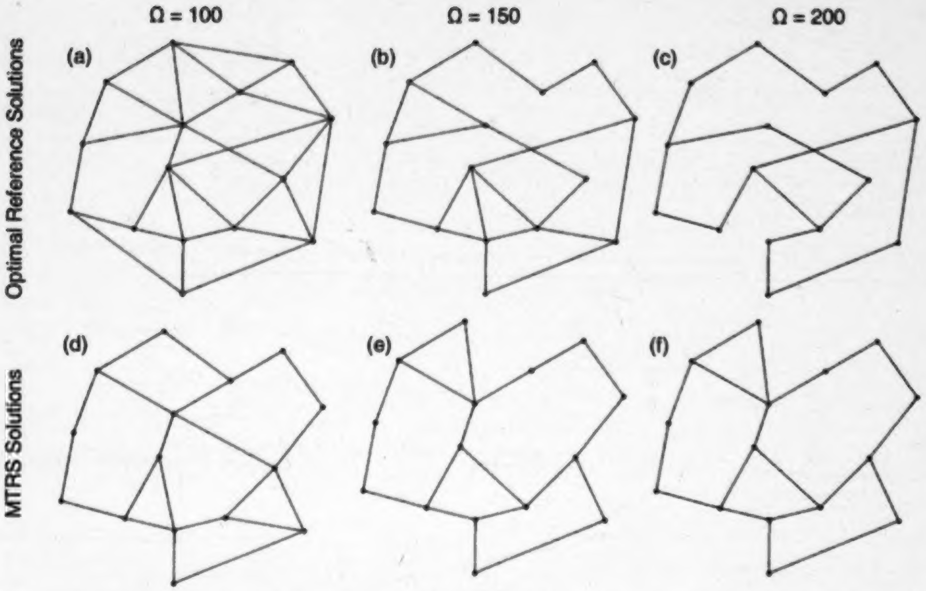

Ω = 100
Ω = 150
Ω = 200
Optimal Reference Solutions
(a)
(b)
(c)
MTRS Solutions
(d)
(e)
(f)

Figure 8.12 – Optimal MTRS and reference solution network graphs for 15n30s1-based designs with three different edge-to-unit-capacity cost ratios.

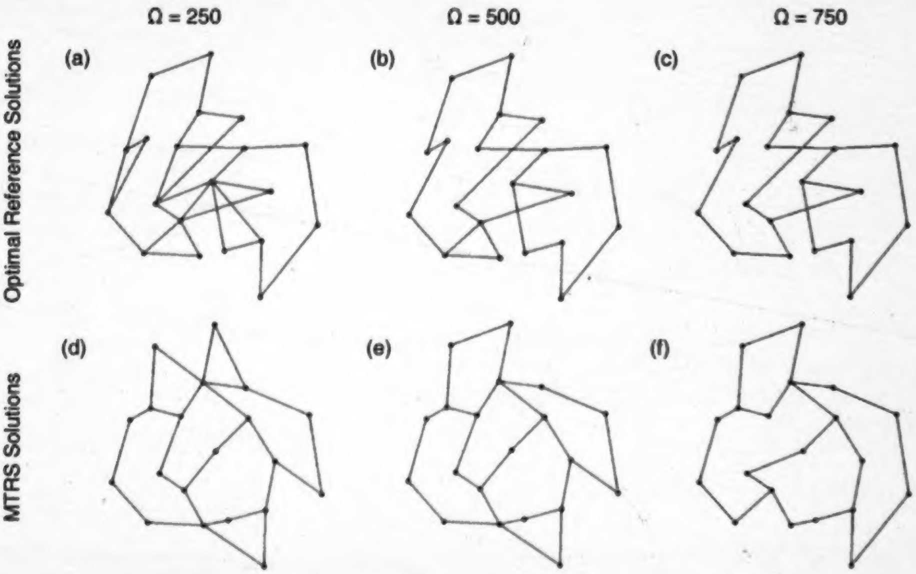

Ω = 250
Ω = 500
Ω = 750
Optimal Reference Solutions
(a)
(b)
(c)
MTRS Solutions
(d)
(e)
(f)

Figure 8.13 – Optimal MTRS and reference solution network graphs for 20n40s1-based designs with three different edge-to-unit-capacity cost ratios.

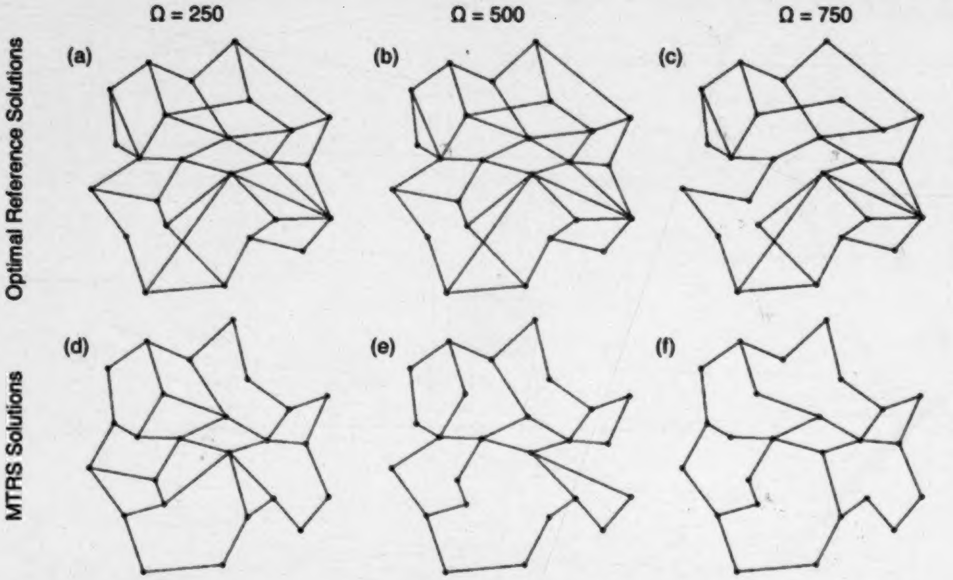

Ω = 250
Ω = 500
Ω = 750
Optimal Reference Solutions
(a)
(b)
(c)
MTRS Solutions
(d)
(e)
(f)

Figure 8.14 – Optimal MTRS and reference solution network graphs for 25n50s1-based designs with three different edge-to-unit-capacity cost ratios.

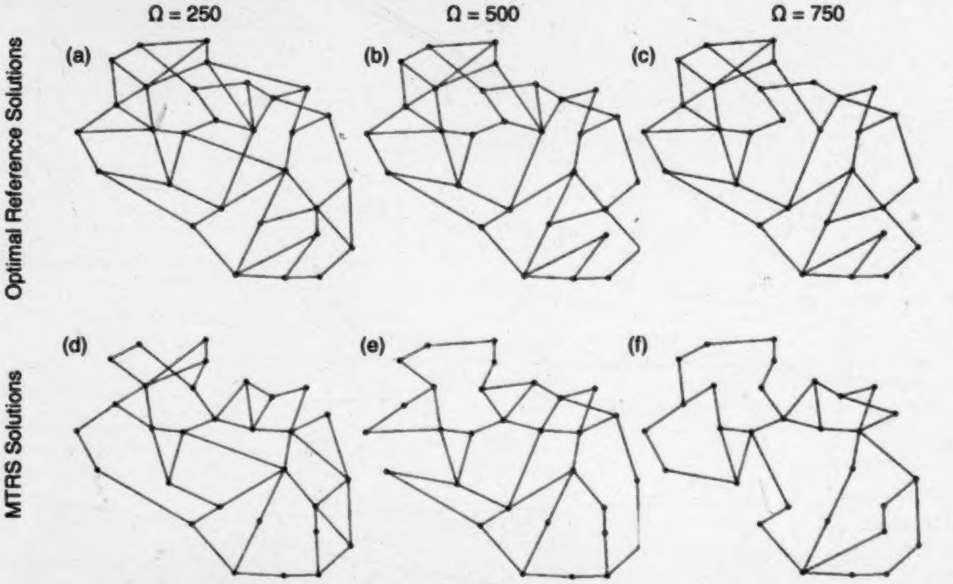

Ω = 250
Ω = 500
Ω = 750
Optimal Reference Solutions
(a)
(b)
(c)
MTRS Solutions
(d)
(e)
(f)

Figure 8.15 – Optimal MTRS and reference solution network graphs for 30n60s1-based designs with three different edge-to-unit-capacity cost ratios.

8.2.2.4 A DISCUSSION ON MTRS COMPLEXITY

The considerable complexity of the MTRS problem can be directly assessed and expressed in terms of the number of variables and constraints involved in the model. In a network with N nodes and S uni-directional edges, the MTRS problem then consists of S edge-selection variables, $\delta_{i,j}$, S working capacity variables, $w_{i,j} \geq 0$, and S spare capacity variables, $s_{i,j} \geq 0$, as well as $S \cdot (S-1)$ restoration flow variables, $s_{i,j}^{k,l} \geq 0$, and assuming all O-D node pairs exchange demand, $S \cdot (N \cdot (N-1)/2) = S^2/2$ working flow variables, $w_{i,j}^{r} \geq 0$. This amounts to a total of $2 \cdot S + 3 \cdot S^2/2$ variables. Adding to the difficulty of the problem is the fact that the S edge-selection variables are all binary (1,0) variables. The number of constraints in the problem is even bigger. Equations (8.14) through (8.17) account for a total of $4 \cdot N \cdot (N-1)$ constraints, equation (8.18) adds another $N \cdot (N-1) \cdot (N-2)$, equations (8.19) through (8.23) and (8.27) through (8.29) add another $8 \cdot S$, equation (8.24) contains $S \cdot N$ constraints, and equations (8.25) and (8.26) add another $2 \cdot S^2$. The total number of constraints is therefore $N^3 + N^2 + N \cdot (S-2) + 8 \cdot S + 2 \cdot S^2$.

So this means that even for our smallest master network (15n30s1), there are 5520 decision variables and 6060 constraints. In the full-mesh version of the network, the problem becomes exponentially bigger since the full-mesh graph of a network with N nodes has a total of $S = N \cdot (N-1)$ uni-directional edges, which is O(N^2). The number of variables in a full-mesh network is therefore O(N^4), as is the number of constraints. In the full-mesh version of the 15n30s1 network, the number of variables and constraints balloons to 66,570 decision variables and 96,600 constraints. While this may be manageable with a very powerful computer, the problem becomes quite intractable for larger networks. In a 40-node full-mesh network, there will actually be 3,653,520 decision variables and 5,007,600 constraints, and the complexity of that problem is practically impossible to solve with the methods and tools available. Clearly this is a problem for which approximations or other simplifying decompositions or heuristics can be justified.

8.3 THREE-STAGE HEURISTIC SOLUTION METHOD

Before looking at a heuristic solution method, we should first understand some of the counteracting effects involved in solution of the MTRS problem. We've already seen in CHAPTER 6 that the total amount of spare capacity required for full network restoration decreases with increases in the topology's average nodal degree, and that it generally follows a $1/(\bar{d}-1)$ form of reduction in spare capacity cost. The $1/(\bar{d}-1)$ lower bound on the redundancy required for survivability is based purely on topological considerations. This is also the basis for intuition that the capacity-efficiency of a mesh-restorable network is greater on highly connected graph topologies. The main observations that we can make regarding the benchmarking trials of CHAPTER 6 that have some relevance here are that the spare capacity requirements drop more quickly with increases in $\bar{d}$ than do working capacity requirements, and working capacity requirements level out much more quickly than do spare capacity requirements, which continue decreasing even at much higher $\bar{d}$. We suspect that this is because restoration routing benefits not only from a greater diversity of eligible restoration routes, but also from the decrease in working capacity itself. These topological effects on spare capacity provide an economic push towards high graph connectivity.

On the other hand, every span included in the final topology will contribute its fixed-charge establishment cost to the total network cost. The direct contribution to the cost made by the network's topology is therefore proportional to the graph connectivity. The minimum-cost spanning tree represents the least investment in span establishment costs that allows communication between any two nodes, but trees are not restorable so the corresponding entity for a mesh-survivable network (in the sense of minimum edge costs, all nodes connected, and restorable) is a minimum-cost bi-connected graph.

Working path routing and working capacity requirements are also in favour of greater graph connectivity. Each additional edge admitted in the graph will permit a shortened routing for some working lightpaths, potentially freeing transmission capacity on a number of spans. The shortening of working routes should continue to be a significant principle in determining an optimum fully restorable topology because, as the network becomes more connected, the working *and* spare capacity requirements diminish. Eventually, increases in graph connectivity reach the point where the working capacity costs dominate over the spare capacity costs, and so it is actually reductions in working capacity costs that we stand to benefit more from (since a given percentage decrease in spare capacity cost is not worth as much in absolute terms as the same percentage decrease in working capacity costs).

So we might surmise that, although both working and spare capacity benefit from increasing connectivity, the absolute potential for payback shifts increasingly from spare capacity savings to working capacity savings. This line of reasoning influences the topology design strategy that follows in that it suggests that a basic priority might be to first design for efficient working-path routing, and only secondly design for efficient spare capacity adapted to the topology from the first priority.

Based on the above considerations, we propose the following three-step heuristic solution method for solving the MTRS problem.

- ***Step W1:*** Solve a fixed charge plus routing (FCR) problem to identify edges that are collectively well suited for routing of working flows. By virtue of their key role in serving working demand flows, we expect that these edges are also of high merit for consideration in a complete MTRS design.
- ***Step S2:*** Solve a new ILP problem for the minimum cost of additional edges and spare capacity to ensure full restorability of the working flows from step W1. We call this the *reserve network fixed charge and sparing* (RN-FCS) problem. Additional edges identified by this step are collectively well suited to enable restoration. By virtue of their efficiency from a restoration standpoint, we expect that they would be of high merit for special consideration in a complete MTRS design. Any edges identified in step W1 are represented by asserting their existence through $\delta_{i,j}=1$ equality constraints on their edge decision variables, and setting their edge establishment costs to zero ($F_{i,j}=F_{j,i}=0$).
- ***Step J3:*** Solve a restricted instance of the MTRS problem, where the set of candidate edges includes only the sets of edges selected for existence in

steps W1 and S2, rather than the complete set of possible edges for the original unrestricted MTRS problem. The idea is that since the MTRS problem is exponential in $|S|$, there is very high run-time leverage on reducing the number of candidate edges. We expect that since the reduced set of candidate edges have already been identified as being of high merit from the standpoint of working routing in step W1 and/or restoration routing in step S2 in isolation, then the solution quality should not suffer too greatly. Solution of this restricted instance of the full problem will then re-optimize the working and restoration routing and select a final set of edges collectively well suited for both aspects of the problem.

The central hypothesis of the heuristic is that within the union of the edge sets arising from steps W1 and S2, there will exist a sub-graph on which a high-quality approximation to the complete MTRS problem can be found. The computational advantage of this should be quite significant. While the FCR in step W1 is by itself quite a difficult problem, its use in the heuristic is to identify high merit edges for the restricted MTRS problem in step J3, and so there is no real need for it to be solved to strict optimality. As such, the solution of step W1 can be a partially relaxed and/or time-limited instance of the FCR problem. Additionally, since FCR is such a widely studied problem, there is a considerable body of prior work available for improving its solution. The problem in step S2, on the other hand, generally solves quite quickly by comparison, mainly because it is composed of a set of non-simultaneous single-commodity network flow sub-problems, whereas the FCR problem in step W1 has a simultaneous multi-commodity transportation-like problem structure. Also, the RN-FCS problem in step S2 only has to consider on the order of N individual span failure scenarios for restoration routing (one for each edge selected in step W1, which will typically comprise a tree-like structure of only slightly more than N edges), while the FCR problem in step W1 has up to $N \cdot (N-1)/2$ demands requiring working routing. Finally, step J3 should be exponentially faster to solve than the complete unrestricted MTRS problem because of the greatly reduced set of candidate edges.

Although the solution obtained from the heuristic will be sub-optimal in the global sense, it is still an exact solution in the sense that the step J3 problem is an un-relaxed and strictly optimal solution of an MTRS problem. The difference between the problem in step J3 and the original and complete MTRS problem is analogous to the difference between a conventional span-restorable SCA problem with only a small set of eligible restoration routes and one where every possible restoration route has been enumerated. In both cases, the solution will exactly specify a fully feasible (and efficient) solution with selection of a set of edges to support full working and restoration routing and proper working and spare capacity allocations to accommodate the resultant working and restoration flows. In other words, there are no functional or constructional details that are approximations of the original problem and have to be repaired as a result of obtaining the design by the heuristic.

The heuristics from the Zoom-In approach [93] are complementary to the idea and approach that follows here. The main difference is in the style of approach. Zoom-In is based on an algorithmic search on topology and a suite of sub-tools that may or may not all be used on a given problem or at a given stage in its refinement. These are strengths for application in network planning software. In contrast, what we explore now is more of a specific hypothesis about the underlying structure of the MTRS problem and attempts to use MIP type solution tools throughout to find a high quality design without an explicit algorithmic search. Our aspiration is to provide a hopefully insightful, but relatively specific tactic for decomposition of the topology, routing, and sparing problem. To the extent that the following heuristic captures a valid insight about the assembly of a "good" topology for MTRS, it may be seen as an additional tactic to propose topology within a larger search strategy. It seems likely that there are ways in which elements of the basic Zoom-In approach and the present method could be combined in future work.

We now more fully describe each step of the proposed heuristic in more detail.

8.3.1 STEP W1 – WORKING-ONLY FIXED CHARGE PLUS ROUTING

The first step of the heuristic is to solve an instance of the FCR problem within the universe of all possible edge selections. We make no regard for any survivability considerations, and so the problem here is identical to that from Section 8.1.1. The only information we need going forward from this point is the set of edges selected (those with $\delta_{i,j}=1$) and the working capacity values needed to support the working routing. The detailed routing associated with the FCR solution here will not be retained or used for any subsequent steps, so the working flow variables are candidates for relaxation to speed up this step. The general idea is only to produce an initial topology and capacity allocation for which an efficient routing solution exists, where the solution would be nearly optimal if the overall goal was to serve the working demand flows only.

We expect this to be a good foundation for the heuristic solution because as discussed earlier in this chapter, working capacity is expected to dominate over the spare capacity, and so it is important to find a preliminary set of edges that is capable of reducing working capacity requirements as much as possible. However, there is nothing in the FCR problem that will assure that a restorable two-connected topology will emerge, and in fact, a spanning tree may even emerge at this stage. Step W1 benefits computationally relative to the complete MTRS problem by way of removing all restoration-related constraints and variables, by permitting full relaxation of all of the working flow variables, and allowing a time-limited or otherwise suboptimal solution to be adequate. This latter reduction is acceptable because the final solution in step J3 doesn't require a strictly optimal solution at this stage, but rather one that merely suggests high-merit candidate edges. In any case, step W1 still remains the most complex stage of the three-step method.

8.3.2 STEP S2 – RESERVE NETWORK FIXED CHARGE AND SPARING

The result from W1 is an initial interim topology and set of working capacity values on those edges, so as to fully serve the demand matrix at minimum cost. The ILP problem in step S2 then augments that topology solution to create a restorable (i.e., at least two-connected) topology while simultaneously minimizing the fixed charge costs of new edges added at this step and the cost of spare capacity required for full restorability of all the working capacities from the working demand routing of step W1. The topology from W1 is now accepted as a set of edges that already exist and for which no establishment cost is charged in step S2 (we assert their existence through $\delta_{i,j}=1$ equality constraints and set $F_{i,j}=F_{j,i}=0$, and only those edges are considered for failure scenarios. From a restoration flow standpoint, however, all possible edges are considered, and the same fixed charges that applied in W1 will still apply for all edges not already selected for existence by step W1. Thus, new edges will be added to the topology by step S2 if they are justified on their combined merits of closing the graph and providing the best placement of restoration capacity.

Relative to the full MTRS problem, step S2 will benefit computationally in three ways:

1. All working flow and capacity variables and constraints are eliminated.
2. Not all $|S|$ possible span failure scenarios have to be considered, but only slightly more than $|N|-1$ as determined by the existing edges from the W1 solution. Since $|S|$ is $O\left(|N|^2\right)$, this reduction to $O(|N|)$ will mean a quite significant reduction in variables and constraints.
3. The edge decision variables are reduced from $|S|$ down to no more than $|S|-|N|+1$ remaining edge choices because at least $|N|-1$ edges were already selected by step W1. Even though $|N|$ may be small compared to $|S|$, there is a more than proportional benefit to the run times because the commitment to inclusion of the sub-graph from W1 greatly reduces the number of remaining search nodes for a branch and bound type solver.

The ILP model for the RN-FCS problem makes use of all the previous notation defined for the FCR and MTRS problems as well as one new set.

New Set:

- $S^{W1} \subseteq S$ is the set of all directional edges in the network for which $\delta_{i,j}=1$ in the FCR solution to step W1. Like the full set of spans, *S*, S^{W1} is typically indexed by the pair *i,j*, which represents the directional span from node *i* to node *j*, and directionality is only used to facilitate expression of the transhipment constraints.

The formulation itself is expressed as follows:

$$\text{Minimize} \quad \sum_{\forall i,j\in S | i,j\notin S^{W1}} \left(c_{i,j}\cdot s_{i,j} + F_{i,j}\cdot \delta_{i,j}\right) + \sum_{\forall i,j\in S^{W1}} \left(c_{i,j}\cdot s_{i,j}\right) \qquad (8.32)$$

$$\text{Subject to:} \quad \sum_{\forall i,k\in S | i\neq k} s_{i,j}^{i,k} = w_{i,j} \qquad \forall i,j\in S^{W1} \qquad (8.33)$$

$$\sum_{\forall k,i\in S} s_{i,j}^{k,i} = 0 \qquad \forall i,j\in S^{W1} \qquad (8.34)$$

$$\sum_{\forall k,j\in S | j\neq k} s_{i,j}^{k,j} = w_{i,j} \qquad \forall i,j\in S^{W1} \qquad (8.35)$$

$$\sum_{\forall j,k\in S} s_{i,j}^{j,k} = 0 \qquad \forall i,j\in S^{W1} \qquad (8.36)$$

$$\sum_{\forall n,k\in S | k\notin\{i,j\}} s_{i,j}^{n,k} - \sum_{\forall k,n\in S | k\notin\{i,j\}} s_{i,j}^{k,n} = 0 \quad \forall i,j\in S^{W1} \quad \forall n\in N | n\notin\{i,j\} \qquad (8.37)$$

$$s_{k,l} \geq s_{i,j}^{k,l} \qquad \forall i,j\in S^{W1} \quad \forall k,l\in S | k,l\neq\{i,j\} \qquad (8.38)$$

$$s_{k,l} \geq s_{j,i}^{k,l} \qquad \forall i,j\in S^{W1} \quad \forall k,l\in S | k,l\neq\{i,j\} \qquad (8.39)$$

$$s_{i,j} \leq K\cdot\delta_{i,j} \qquad \forall i,j\in S | i,j\notin S^{W1} \qquad (8.40)$$

$$\delta_{i,j} = \delta_{j,i} \qquad \forall i,j\in S | i,j\notin S^{W1} \qquad (8.41)$$

$$\delta_{i,j} \leq 1 \qquad \forall i,j\in S | i,j\notin S^{W1} \qquad (8.42)$$

$$\delta_{i,j} = 1 \qquad \forall i,j\in S^{W1} \qquad (8.43)$$

The objective function in equation (8.32) is very similar to the objective function in equation (8.13) for the MTRS formulation, with the only difference being that we now remove all of the $w_{i,j}$ working capacity variables. The constraints in equations (8.33) through (8.37) are the same as the corresponding restoration flow transhipment constraints for the MTRS formulation, except that we now consider only spans in S^{W1} as failure scenarios. The spare capacity dimensioning constraints in equations (8.38) and (8.39) are also repeated from the MTRS formulation except that they too now only consider the reduced set of failure scenarios. In equation (8.40), the edge decision variables on all spans not yet selected in the FCR solution of step W1 are forced to be $\delta_{i,j}=1$ if spare capacity is assigned to them, which then contributes those edges' fixed charges to the objective function. The constraints in equations (8.41) and (8.42) are the same as those in equations (8.28) and (8.29) in the MTRS formulation except that they also now apply only to the spans $i, j \in S \mid i, j \notin S^{W1}$.

In addition, optional constraints in equations (8.30) and (8.31) requiring a minimum of $|N|$ edges and degree-2 for all nodes, respectively, are not logically required, but can speed up the branch-and-bound solution if we include them in the model.

8.3.3 STEP J3 – RESTRICTED MTRS FOR FINAL TOPOLOGY AND CAPACITY

The final step of the heuristic is to solve a reduced instance of the MTRS problem, where the set S is reduced to include only those spans for which $\delta_{i,j} = 1$ in the RN-FCS solution of step S2. This will address the global co-ordination of working, spare and topology considerations that are inherent in the full problem but not present in the design at the end of S2. Because the set of edges is so much smaller here than in the complete problem, solution runtimes are expected to be reduced significantly (and more than proportionally since many variables and constraints are $O\left(|S|^2\right)$).

The resulting solution to step J3 will only retain or possibly even further reduce the edge set from S2, and since those edges were already judged to be of high merit for working routing (step W1) and restoration (step S2), we expect this solution to be near optimal.

In summary, the three steps play the following roles:

- Step W1 finds a minimal topology and capacity as justified by working flows alone.
- Step S2 finds a minimum-cost topology augmentation as justified by restorability considerations alone.
- Step J3 revises the working flows of W1 to exploit the augmented topology of S2 and coordinates them with the assignment of restoration capacity and selection of edges so as to minimize the total network cost.

8.3.4 ADDITIONAL CONSIDERATIONS AND OPTIONAL BOUNDS

In cases where the set of candidate edges provided to step J3 is quite small, or where solution of step J3 is exceptionally fast, we might repeat steps W1 and S2 and use an artificially low edge-to-unit-capacity cost ratio parameter ($\Omega = F_{i,j}/c_{i,j}$) in those steps. This will shift the emphasis of the objective function to favour solutions with more efficient working and restoration routing, rather than those with fewer edges. So the outcome at those stages will be a slightly more richly connected network topology, which ultimately will provide a somewhat larger subset of candidate edges for the step J3 solution, and a potentially more efficient final solution.

The objective function value from step W1 can also be used as an added lower bound on the objective function in step J3 to aid in its solution (that is, if step W1 was solved to optimality). Since the FCR solution of step W1 represents the lowest possible cost network that serves all working demands, any feasible solution for fixed charges, routing and spare in step J3 must at least cost as much as the optimal solution for FCR alone.

Similarly, the objective function value from step W1 can be used as a lower bound on the objective function for the complete MTRS problem as well when attempting to solve for an optimal reference solution. An even tighter lower-bound can be identified here that applies on a sub-set of the variables in the MTRS objective function by applying the same line of reasoning to the *topology plus working capacity variables only* within an MTRS problem. In other words, we can say that the objective function value from W1 is a lower bound on $\sum_{\forall i,j \in S} \left(c_{i,j} \cdot w_{i,j} + F_{i,j} \cdot \delta_{i,j} \right)$ in the complete problem.

This is because the fixed charge and working routing solution alone that is embodied within a full MTRS solution can only make compromises to accommodate the wider set of considerations in MTRS compared to the pure FCR solution from W1.

In addition, the objective function value of the step J3 solution can be used as an upper bound in the complete MTRS problem. Because the complete MTRS problem and step J3 are identical models, but with step J3 simply being restricted in the set of edges to consider, the objective function from step J3 must be an upper bound on the corresponding instance of the full MTRS problem solved to optimality.

8.4 EXPERIMENTAL STUDY METHOD

To validate the three-step heuristic, characterize the effects of varying edge-to-unit-capacity cost ratios, $\Omega = F_{i,j}/c_{i,j}$, and other parameters such as deliberately time-limiting steps W1 and S2, and compare total runtimes and objective function values to complete MTRS problem solutions, we conducted comparative capacity design trials using the same test-case network topologies as the MTRS tests in Section 8.2.2.

As with the complete MTRS tests, the steps W1, S2, and J3 models were all implemented in AMPL Mathematical Programming Language version 9.0 and solved with CPLEX 9.0 MIP Solver running on a 4-processor SUN UltraSparc III running at 900 MHz with 16 GB of RAM. The optional constraints (e.g., equations (8.30) requiring a minimum of $|N|$ edges in step S2, etc.) are all used as applicable. Pre-processing for eligible working and restoration routes was not required, but data files containing network topology information and other input parameters were prepared on a dual-processor AMD Opteron 242 PC with 1 GB of RAM, running Windows 2000. All working and spare capacity allocations were integer, corresponding to capacity design and restoration mechanisms at the wavelength level, and edge decision variables, $\delta_{i,j}$, were strictly binary. Because of the complexity of the W1, S2, and J3 ILP models, most results are based on time-limited CPLEX runs with various MIPGAP settings ranging from 10^{-2} (i.e., solutions are guaranteed to be within 1% of optimal) to 0.20 (i.e., a fully terminated solution would have an optimality gap of 20%). The recorded runtimes are actual elapsed time, which is roughly equivalent to actual CPU time on the four-processor unit as a whole, where all four processors are devoted to the CPLEX task full time. As with the results in Section 8.2.2, actual runtimes and optimality gaps are provided in the appropriate tables that follow.

8.5 RESULTS AND DISCUSSION

Table 8.4 summarizes the results of our tests of the three-step heuristic. The first three columns indicate the test number, the test network, and the Ω value for that particular test case. The next set of four columns under the "MTRS (J0)" heading provides the solution to the full MTRS problem that we obtained in Section 8.2.2. This is the benchmark solution to which we will compare the three-step heuristic results, and we can call it the "J0" solution to denote the fact that it is a jointly optimized benchmark solution. The sets of columns under the "W1", "S2", and "J3" headings provide the results of the first, second, and third steps of the heuristic, respectively. In the W1 columns, the Ω value here corresponds to the one used for the W1 step only, which for reasons to be discussed later, we may wish to be different than the Ω value we use for the J0 and J3 solutions. The other four columns in the W1 step tell us how many spans the W1 solution contained, what its cost is (the sum of the total working capacity and fixed charges of spans used), how long it took to solve, and what the maximum gap to optimality is. The columns under the "S2" provide the same data for the S2 reserve network fixed charge plus routing solution. Again here, we may wish to use a different Ω value from the one we use for the J0 and J3 solutions. The number of spans in the S2 solution includes those inherited from the W1 solution as well as those additional spans needed to close the graph from W1 and optimize its restoration routing. We also add an extra column here to show how many new spans have been added by the S2 step. The cost of the S2 solution given here corresponds to the total cost of working and spare capacity, plus the fixed charge costs of all spans used in the solution. In the "J3" columns, we show the number of spans in the final fully optimized J3 solution, the number of spans from the S2 solution that are not a part of the J3 solution, the cost of the J3 solution, the runtime required to solve it, the maximum gap to optimality, and the objective function improvement relative to the benchmark J0 solution, respectively. In the latter column, a negative value indicates that the J3 solution was more costly than the J0 reference solution, and a positive value indicates that the J3 solution was actually better than J0. We will explain shortly how this is possible. In the tests summarized in Table 8.4, only the six master networks are used. Tests on the full-mesh versions of those networks will follow.

Table 8.4 – Heuristic solutions of all six master networks, each with three different edge-to-unit-capacity cost ratios.

			MTRS (J0)				W1					S2						J3					
#	Test Network	Ω	# Spans	Cost	Time (hours)	MIP Gap	Ω	# Spans	Cost	Time (min)	MIP Gap	Ω	# Spans	New Spans	Cost	Time (sec)	MIP Gap	# Spans	Elim. Spans	Cost	Time (min)	MIP Gap	% Improv.
1	15n30s1	100	21	0.83	0.4	1.0%	100	14	0.54	0.25	1.0%	100	19	5	1.02	1.0	1.0%	19	0	0.87	0.04	1.0%	-4.7%
2	15n30s1	150	19	1.02	0.6	1.0%	150	14	0.66	0.34	1.0%	150	19	5	1.20	0.4	1.0%	19	0	1.05	0.03	1.0%	-2.9%
3	15n30s1	200	19	1.20	0.5	1.0%	200	14	0.79	0.21	1.0%	200	19	5	1.37	0.4	1.0%	18	1	1.23	0.05	1.0%	-2.0%
4	20n40s1	250	26	0.76	8.3	5.0%	250	20	0.50	2.32	1.0%	250	26	6	0.85	2.2	1.0%	25	1	0.78	0.31	1.0%	-2.7%
5	20n40s1	500	25	1.16	8.7	5.0%	500	19	0.78	2.96	1.0%	500	26	7	1.32	1.9	1.0%	26	0	1.19	0.61	1.0%	-2.6%
6	20n40s1	750	23	1.55	8.6	5.0%	750	19	1.04	1.74	1.0%	750	24	5	1.74	1.5	1.0%	23	1	1.58	0.15	1.0%	-2.1%
7	25n50s1	250	36	0.85	3.4	10.0%	250	24	0.56	7.0	1.0%	250	35	11	1.03	4.6	5.0%	33	2	0.86	6.2	5.0%	-1.0%
8	25n50s1	500	32	1.26	6.0	13.6%	500	24	0.83	4.6	1.0%	500	31	7	1.42	2.6	5.0%	31	0	1.25	1.6	5.0%	0.9%
9	25n50s1	750	31	1.59	6.0	10.3%	750	24	1.09	5.7	1.0%	750	32	8	1.78	2.1	5.0%	32	0	1.61	6.6	5.0%	-1.7%
10	30n60s1	250	46	0.92	7.7	20.0%	250	30	0.58	28.6	1.0%	250	40	10	1.11	3.0	5.0%	40	0	0.94	1.6	5.0%	-2.1%
11	30n60s1	500	42	1.33	5.9	20.0%	500	29	0.79	21.0	1.0%	500	40	11	1.52	8.8	5.0%	39	1	1.25	7.8	5.0%	6.1%
12	30n60s1	750	38	1.56	6.0	20.0%	750	29	1.00	12.0	1.0%	750	38	9	1.83	6.4	5.0%	38	0	1.57	1.1	5.0%	-0.7%
13	35n70s1	500	64	1.03	12.0	28.9%	500	37	0.55	74.1	5.0%	500	45	8	0.94	7.9	10.0%	45	0	0.86	5.7	10.0%	16.7%
14	35n70s1	750	68	1.43	12.0	37.8%	750	36	0.70	83.0	5.0%	750	45	10	1.20	6.8	10.0%	45	0	1.09	4.3	10.0%	23.8%
15	35n70s1	1000	66	1.72	12.0	38.8%	1000	35	0.84	96.9	5.0%	1000	44	9	1.38	5.5	10.0%	44	0	1.31	10.8	10.0%	24.1%
16	40n80s1	500	78	1.08	12.0	31.4%	500	56	0.70	120.0	21.6%	500	65	9	1.04	2.5	10.0%	65	0	0.93	120	17.6%	14.3%
17	40n80s1	750	78	1.41	12.0	36.8%	750	75	1.23	120.0	47.4%	750	76	1	1.48	1.9	10.0%	--- Infeasible ---			120	-	-
18	40n80s1	1000	79	1.78	12.0	20.0%	1000	79	1.61	120.0	51.6%	1000	79	0	1.83	0.9	10.0%	--- Infeasible ---			120	-	-
19	40n80s1	500	78	1.08	12.0	31.4%	Inf.	43	0.69	1.31	20.0%	500	53	10	1.28	12.3	10.0%	53	0	0.91	6.7	10.0%	16.2%
20	40n80s1	750	78	1.41	12.0	36.8%	Inf.	43	0.82	1.31	20.0%	750	53	10	1.48	11.2	10.0%	53	0	1.10	13.6	10.0%	21.6%
21	40n80s1	1000	79	1.78	12.0	20.0%	Inf.	43	0.96	1.31	20.0%	1000	48	5	1.65	12.4	10.0%	48	0	1.33	15.3	10.0%	25.6%
2a	15n30s1	150	19	1.02	0.6	1.0%	100	14	0.54	0.25	1.0%	100	19	5	1.02	1.0	1.0%	19	0	1.06	0.03	1.0%	-2.9%
3a	15n30s1	200	19	1.20	0.5	1.0%	150	14	0.66	0.34	1.0%	150	19	5	1.20	0.4	1.0%	19	0	1.26	0.05	1.0%	-5.1%
3b	15n30s1	200	19	1.20	0.5	1.0%	100	14	0.54	0.25	1.0%	100	19	5	1.02	1.0	1.0%	19	0	1.26	0.05	1.0%	-5.1%
5a	20n40s1	500	25	1.16	8.7	5.0%	250	20	0.50	2.32	1.0%	250	26	6	0.85	2.2	1.0%	25	1	1.17	0.80	1.0%	-0.9%
6a	20n40s1	750	23	1.55	8.6	5.0%	500	19	0.78	2.96	1.0%	500	26	7	1.32	1.9	1.0%	26	0	1.74	0.66	1.0%	-12.1%
6b	20n40s1	750	23	1.55	8.6	5.0%	250	20	0.50	2.32	1.0%	250	26	6	0.85	2.2	1.0%	26	0	1.72	1.08	1.0%	-10.8%
8a	25n50s1	500	32	1.26	6.0	13.6%	250	24	0.56	7.0	1.0%	250	35	11	1.03	4.6	5.0%	35	0	1.32	18.0	5.0%	-4.5%
9a	25n50s1	750	31	1.59	6.0	10.3%	500	24	0.83	4.6	1.0%	500	31	7	1.42	2.6	5.0%	31	0	1.61	3.5	5.0%	-1.3%
9b	25n50s1	750	31	1.59	6.0	10.3%	250	24	0.56	7.0	1.0%	250	35	11	1.03	4.6	5.0%	35	0	1.76	40.5	5.0%	-10.9%
11a	30n60s1	500	42	1.33	5.9	20.0%	250	30	0.58	28.6	1.0%	250	40	10	1.11	3.0	5.0%	40	0	1.28	5.7	5.0%	3.2%
12a	30n60s1	750	38	1.56	6.0	20.0%	500	29	0.79	21.0	1.0%	500	40	11	1.52	8.8	5.0%	40	0	1.69	10.2	5.0%	-8.3%
12b	30n60s1	750	38	1.56	6.0	20.0%	250	30	0.58	28.6	1.0%	250	40	10	1.11	3.0	5.0%	40	0	1.65	9.7	5.0%	-5.7%
14a	35n70s1	750	68	1.43	12.0	37.8%	500	37	0.55	74.1	5.0%	500	45	8	0.94	7.9	10.0%	45	0	1.08	3.9	10.0%	24.0%
15a	35n70s1	1000	66	1.72	12.0	38.8%	750	35	0.70	83.0	5.0%	750	45	10	1.20	6.8	10.0%	45	0	1.29	6.2	10.0%	25.0%
15b	35n70s1	1000	66	1.72	12.0	38.8%	500	37	0.55	74.1	5.0%	500	45	8	0.94	7.9	10.0%	45	0	1.29	8.8	10.0%	25.1%
17a	40n80s1	750	78	1.41	12.0	36.8%	500	56	0.70	120.0	21.6%	500	65	9	1.04	2.5	10.0%	65	0	1.19	120	23.9%	15.6%
18a	40n80s1	1000	79	1.78	12.0	20.0%	750	75	1.23	120.0	47.4%	750	76	1	1.48	1.9	10.0%	--- Infeasible ---			120	-	-
18b	40n80s1	1000	79	1.78	12.0	20.0%	500	56	0.70	120.0	21.6%	500	65	9	1.04	2.5	10.0%	65	0	1.44	120	27.8%	19.1%
20a	40n80s1	750	78	1.41	12.0	36.8%	Inf.	43	0.69	1.31	20.0%	500	53	10	1.28	12.3	10.0%	48	5	1.06	77.6	10.0%	24.7%
21a	40n80s1	1000	79	1.78	12.0	20.0%	Inf.	43	0.82	1.31	20.0%	750	53	10	1.48	11.2	10.0%	48	5	1.26	58.3	10.0%	29.5%
21b	40n80s1	1000	79	1.78	12.0	20.0%	Inf.	43	0.69	1.31	20.0%	500	53	10	1.28	12.3	10.0%	47	6	1.24	102.9	10.0%	30.2%

The first observations we make are in regards to the way a network graph evolves as we progress through the three-step heuristic. In most test cases, the W1 solution makes use of only as many spans as are needed to fully connect the network (i.e., $|N|-1$ spans), plus perhaps one or two other spans. For instance, in test cases #1 to #3 (for the 15n30s1 master network), only 14 spans are selected in each case, and in test cases #4 to #6 (20n40s1), 19 spans were used for $\Omega = 750$ and $\Omega = 500$, and 20 were used for $\Omega = 250$. We also note that as expected, the number of spans used in the W1 solution tends to decrease slightly for higher Ω values. As the cost of adding a new span increases, it is more economical to use slightly more working capacity rather than add a new span. In the S2 solutions, the network graphs all tended to have average nodal degrees of approximately $d = 2.6$, and except for test cases #16 to #18, which have very high MIP gaps (and are therefore very far from optimal S2 solutions), they all have $2.4 \le d \le 2.8$. Examining the spans used in the S2 step, we find that they exhibit three types of edge changes relative to the W1 solution. An edge could be added in a way that provides both graph closure *and* bears spare capacity, an edge could be added that just bears spare capacity but does not contribute to closure of the graph (i.e., the graph would still be closed without it), or an edge that was present in the W1 topology is not logically present in the reserve network overlay design of S2. In the latter type, this means that an edge used to route working demands in the W1 solution bears no spare capacity in the S2 solution, but is still present by virtue of it being inherited from the W1 solution (and the fact that it still bears working capacity). Finally, for the J3 solution, which is a reduced instance of the full MTRS problem solved only on the set of spans needed in the W1 and/or S2 solution, we notice that only in four of the 21 test cases, was the set of spans present after the S2 step reduced by the elimination of some span(s). In three of those test cases (#4, #6, and #11), only a single span was disused, and in the other (test case #7), only two were eliminated. The fact that we don't see a greater degree of graph rationalization by J3 suggests that perhaps the set of edges being suggested by the W1 and S2 steps is not diverse enough. In other words, the heuristic may be using too few edges. We will revisit this thought later.

In the smaller test cases (#1 through #9) the final heuristic solutions were an average of 2.1% more costly that the benchmark J0 solutions. Solutions ranged from 4.7% worse in test case #1 for the 15-node master network with $\Omega = 100$ to 0.9% *better* in test case #8 for the 25-node master network with $\Omega = 500$. While it may initially seem contradictory that a sub-optimal heuristic is able to provide a better solution than the full MTRS problem can, we can point out that the J0 solutions are not strictly optimal. We recall from Section 8.2.2 that in order to obtain solutions to the full MTRS problem, we needed to use a relatively high MIP gap to optimality and/or limit the runtime and take the best solution available after that indicated runtime. In test case #8 for instance, the J0 solution has a MIP gap of 13.6%, meaning after the 6-hour runtime, the best solution available was as much as 13.6% above the true optimal. So in this case, the 0.9% improvement of the J3 solution over the J0 solution makes sense. In test cases #10 through #18 (30-node, 35-node, and 40-node networks), the J3 solution was actually an average of 11.7% better than the full MTRS solution, and was as high as 24.1% better (test case #15 for the 35-node network with $\Omega = 1000$).

The three-step heuristic was also able to provide its high-quality solutions significantly faster than the full MTRS problem. For test cases #1 through #15, the full MTRS problem took an average of 69 times longer to solve than the three-step heuristic. In all cases, the heuristic runtimes were dominated by the W1 step, which is as we expected and predicted in Section 8.3. The W1 step is particularly difficult to solve because it is essentially the fixed charge plus routing (FCR) problem, which is known to be very difficult to solve for all but the smallest test case networks. However, because the W1 step is merely intended to suggest spans that are of high merit for working routing and provide working capacities for use in the S2 step (rather than fully assembling a strictly optimal FCR solution), we can terminate this step early or set a higher MIP gap if needed. In test cases #13 to #15 (the 35-node network), we set a MIP gap of 5%, and W1 solutions were obtainable in 74 to 97 minutes. In test cases #16 to #18, we limited the W1 runtime to two hours but this did, however, present an unforeseen problem. Because the FCR problem is so difficult to solve, a 2-hour runtime on the 40-node network resulted in a W1 solution that was so far from optimal that it was practically useless for the purposes required by the heuristic. For test case #17, 75 of the 80 spans in the network were selected for use in the W1 solution, and for test case #18, 79 of the 80 spans were selected. As a result, the J3 problem was only very slightly reduced from the full MTRS problem, and so we were unable to obtain a J3 solution for those two test cases in the 2-hour runtime allotted. If we had allowed a longer runtime in either the W1 or J3 step, we would presumably be able to obtain solutions that are comparable to the sub-optimal J0 solutions for those test cases, but the total runtime of the heuristic would have suffered greatly. One alternative is to solve a slightly modified version of the FCR problem in the W1 step, where we essentially let $\Omega = \infty$ so that the solution simply represents the lowest cost set of spans that fully connects the network. We did just that in test cases #19 to #21, where the W1 step could be solved to within 20% of optimal in just slightly more than one minute.

In contrast to the W1 step, the S2 step solved very quickly, requiring only a few seconds to solve even in the worst case. The reason this step was so much easier to solve is, in part, because we effectively have only $\left|S^{W1}\right| \cong |N|$ routing problems to solve (where S^{W1} is the set of spans selected by the W1 solution). The W1 problem, on the other hand, has $|N| \cdot (|N|-1)/2$ routing problems, one for each demand relation. The J3 step also solved quite quickly (with the exception of test cases #16 to #18), and only needed several minutes or less in most cases. In fact, the J3 step solved in under a minute for the 15-node and 20-node test cases, and in 10.8 minutes or less for all of the 35-node or smaller test cases.

We return now to an earlier observation that in most test cases, the J3 solution uses all edges promoted for its consideration by prior steps W1 and S2. This was somewhat unexpected as it was thought initially that the set union of edges from W1 and S2 would tend to over-populate the candidate edge set, leading to a reduction or rationalization of the graph topology by the J3 step. Related to this is the observation that, when the J0 reference solution is not improved upon by the heuristic (i.e., the "% Improv." column has a negative value), the J0 reference solutions consistently use more edges than the heuristic solutions. This suggested an aspect of test cases #2a through #21b, where we thought we might be able to deliberately inflate the cardinality of the edge set promoted by steps W1 and S2 by artificially lowering the Ω

value used in those first two steps. The tactic does work in terms of promoting more edge candidates in the pool for J3, but the final network design solutions of these additional test cases proved to not be as good as we would have expected. In most cases, the corresponding J3 solutions still elect to use exactly the same number of edges as before, but in many cases, the objective value is slightly worse. For instance, for the smaller test cases (the 15-node, 20-node, and 25-node networks), the heuristic solutions were an average of 6.1% more costly than the J0 solutions, compared to 2.1% more costly when we didn't artificially lower Ω. For the 30-node, 35-node, and 40-node test cases, using lower Ω values in the W1 and S2 steps did appear to have a beneficial effect, but it was quite small. Those J3 solutions improved on the benchmark solutions by an average of 16.6%, compared to an improvement of 14.6% when Ω values are not artificially lowered in W1 and S2. In general, however, as we reduce the Ω values for the W1 and S2 steps, we see the J3 objective values worsen. This was an initially unexpected effect. The thinking was that if we bias the W1 and S2 stages to an artificially low Ω value, we would simply qualify more edges for J3 to consider, and that doing so could increase run time slightly but should only improve the achievable solution quality. This was partly also motivated by observation that the J3 problems were solved extremely quickly, compared to W1, so it seemed practical to give J3 more edges and invest more run time at J3 to improve solution quality.

But evidently there are counter-acting effects. One is that even a slight increase in the number of edges offered to J3 can make its run time take off exponentially. In test case #8, for example, the J3 solution took 1.6 minutes to solve when there were 31 edges to consider. By lowering the Ω value in steps W1 and S2 from $\Omega = 500$ to $\Omega = 250$ (in test case #8a), we promoted four additional edges for a total of 35 to be considered in the J3 step, and runtime increased to 18 minutes. Similarly, test case #9 had 32 edges going into J3 and took 6.6 minutes to solve, while test case #9b had an inflated pool of 35 edges and it took 40.5 minutes to solve. Thus, as happened here, providing the wrong set of extra edges for the final step may impact runtime, possibly even to the point of requiring a runtime limit, and obviously this can hurt the solution quality. But more fundamentally, even if run time limits are not involved, the W1 and S2 edges identified at any particular Ω value are well suited to that Ω value and are not necessarily as well suited for a different Ω value. Ironically, this is consistent with the basic hypothesis that W1 and S2 would identify intrinsically good edges for their own purposes at the given Ω values. So we can indeed promote more edges by lowering Ω, but that set does not necessarily contain the same set of high-merit edges we would have at the higher Ω value.

Detailed comparison of the edges used in test case #9 versus those of test case #9b tends to confirm this, and shows how sensitive the overall solution is to slight changes in the eligible pool of edges for J3 to consider. In test case #9, we used $\Omega = 750$ for all three steps, and J3 was given a pool of 32 edges to consider. The resulting solution used all 32 of those spans at a cost of 1.61. In test case #9b, on the other hand, we used $\Omega = 250$ for steps W1 and S2, and J3 was given a pool of 35 edges, 29 of which are in the previous set of 32 from test case #9 (i.e., only three of the 32 edges from test case #9 are not provided to J3 in step #9b, and we give it an additional six edges to consider as well). Once again, all of the candidate edges were used in the final solution, but now the cost was 1.76. So the loss of those three edges led to a 9.1% *increase* in solution cost, and it took more than 6 times longer to solve. This effect, combined with the prior observations that J3 rarely eliminates more than one or two edges, suggests a somewhat different understanding about the heuristic than at the start. Rather than W1 and S2 nominating a pool of high-merit edges from which J3 will select a subset, it seems more accurate to say that W1 and S2 almost directly assemble a high-merit topology, and J3 makes only minor refinements to the topology as possible under the final co-design of working and restoration routes and capacity. At the same time, these findings and arguments do not completely rule out benefit from the strategy of lowering Ω values. Here the idea was tested only with a small number of lower Ω values, but if one was using the heuristic in an extended study of a single planning problem, a range of tests with a greater variety of Ω values would still be recommended to search for enhancements over the basic procedure. At more moderate levels of Ω value depression, one might still be able to promote some additional edges for J3 consideration, without degrading the suitability of the edge set in the vicinity of the proper Ω value as much as what we've observed in the limited tests we've done here.

With the results in Table 8.5, we investigate the use of the three-step heuristic in the full-mesh versions of our test case networks. Table 8.5 is structured exactly the same as Table 8.4, and summarizes the full MTRS (J0) benchmark solutions as well as the network design solutions provided at each step of the heuristic. We recall from Section 8.2.2 that, except for the 15-node network, near-optimal MTRS solutions are practically impossible to obtain for the full-mesh networks; the MTRS problem could not even yield feasible solutions for any of the full-mesh networks larger than 20 nodes in a full 24 or even 48 hours of runtime. On the other hand, solutions were readily obtainable with the three-step heuristic for even the 30-node full-mesh network (test cases #31 to #33) in slightly more than an hour. In test cases #22 to #24 for the 15-node full-mesh network, the heuristic solution was an average of 5.2% better than the full MTRS benchmark problem, and depending on the Ω value used, was solved in approximately 5 to 6 minutes (it took 12 hours for the benchmark). For the 20-node network, the best feasible solutions to the unrestricted J0 problems were 71.4% to 86.4% more costly that the J3 heuristic result. And in two of the 15-node cases, and in all of the 20-node cases, the J0 solutions were characterized by far too many edges in their best feasible solutions after 12 to 24 hours of runtime. Our interpretation of this repeatedly observed effect is that the J0 problem is "bogged down" in high-weight edge-vector space because there are combinatorially many more high connectivity graphs. Simple discovery of low-weight edge vectors that describe closed connected graphs (where the optimal solutions really lie) is very difficult. Because the full-LP relaxations are so terribly weak as lower bounds, the solver's progress is relatively un-guided. Hence the nodes visited in the branch-and-bound tree tend to be high-weight edge vectors simply because these are statistically much more frequent in the population of all possible closed connected graphs. This is the main insight we offer about why the MTRS problem is so exceedingly difficult to solve by MIP methods.

Table 8.5 – Heuristic solutions of the full-mesh versions of the 15-node, 20-node, 25-node, and 30-node master networks, each with three different edge-to-unit-capacity cost ratios.

			MTRS (J0)				W1					S2						J3					
#	Test Network	Ω	# Spans	Cost	Time (hours)	MIP Gap	Ω	# Spans	Cost	Time (min)	MIP Gap	Ω	# Spans	New Spans	Cost	Time (min)	MIP Gap	# Spans	Elim. Spans	Cost	Time (min)	MIP Gap	% Improv.
22	15n30s1-Full	100	24	0.87	12	20.8%	100	15	0.54	4.31	1.0%	100	20	5	0.94	0.7	10.0%	20	0	0.83	0.01	10.0%	4.2%
23	15n30s1-Full	150	22	1.10	12	23.8%	150	14	0.66	5.64	1.0%	150	18	4	1.18	0.1	10.0%	18	0	1.05	0.01	10.0%	4.9%
24	15n30s1-Full	200	20	1.32	12	24.5%	200	14	0.79	5.94	1.0%	200	20	6	1.41	0.2	10.0%	20	0	1.24	0.04	10.0%	6.6%
25	20n40s1-Full	250	105	2.87	24	81.8%	250	22	0.51	19.62	10.0%	250	28	6	0.84	1.8	20.0%	28	0	0.82	0.19	20.0%	71.4%
26	20n40s1-Full	500	105	5.49	24	84.4%	500	20	0.76	23.01	10.0%	500	26	6	1.24	8.2	1.0%	26	0	1.18	0.22	20.0%	78.5%
27	20n40s1-Full	750	134	11.70	24	90.1%	750	21	1.06	10.66	10.0%	750	26	5	1.57	2.1	1.0%	26	0	1.60	0.11	20.0%	86.4%
28	25n50s1-Full	250	-- Infeasible --		24	-	Inf.	27	0.75	14.01	20.0%	250	42	15	1.39	4.2	20.0%	42	0	0.96	1.51	20.0%	-
29	25n50s1-Full	500	-- Infeasible --		24	-	Inf.	27	1.03	14.01	20.0%	500	36	9	1.82	5.6	20.0%	36	0	1.34	0.60	20.0%	-
30	25n50s1-Full	750	-- Infeasible --		24	-	Inf.	27	1.31	14.01	20.0%	750	34	7	2.14	7.5	20.0%	34	0	1.82	0.73	20.0%	-
31	30n60s1-Full	250	-- Infeasible --		48	-	Inf.	33	0.62	39.5	20.0%	250	43	10	1.13	12.2	20.0%	41	2	0.91	13.0	20.0%	-
32	30n60s1-Full	500	-- Infeasible --		48	-	Inf.	33	0.85	39.5	20.0%	500	42	9	1.47	33.4	20.0%	42	0	1.24	1.3	20.0%	-
33	30n60s1-Full	750	-- Infeasible --		48	-	Inf.	33	1.08	39.5	20.0%	750	41	8	1.80	24.2	20.0%	41	0	1.66	2.9	20.0%	-

In an effort to obtain a solution to the full MTRS (J0) problem that is better than the solution to the three-step heuristic for the full-mesh networks, we added a constraint to the MTRS ILP model that forces the corresponding J3 solution cost as an upper bound on the J0 objective function. In other words, after solving the three-step heuristic for one network at a given Ω value, and obtaining a cost of C_{J3}, we went back to the full MTRS problem for that test case and added the following new constraint:

$$\sum_{\forall i,j\in S}\left(c_{i,j}\cdot\left(w_{i,j}+s_{i,j}\right)+F_{i,j}\cdot\delta_{i,j}\right)\le C_{J3} \tag{8.44}$$

The thought was that this would give the ILP solver a head start in assembling a suitable graph topology and we hoped that a near-optimal solution might then be possible. Without that upper bound, the solver was at least able to find a *feasible* solution to the 15-node and 20-node full-mesh networks relatively easily (typically several hours), but when the upper bound was added, the full MTRS problem would no longer even return a feasible solution in the allotted runtime (24 hours). We can also notice that for the full-mesh networks, J3 employed all edges provided to it from W1 and S2, seeming to further suggest that the first two heuristic steps may not be promoting a large enough set of edges to consider in J3. With this understanding, it suggests that without the upper bound in equation (8.44), the attempt at an optimal solution is still searching in high-connectivity topology space after 12 to 24 hours. With the bound, however, it has not even stumbled upon a closed topological arrangement similar to the heuristic's after the 12 to 24 hours of search. In contrast the heuristic, by its nature, is directly guided into a region where topologically feasible arrangements of a not-excessive number of edges are immediately at hand.

The fact that the unrestricted problem fails to even reach feasibility when the J3 objective value is supplied as an upper bound suggests that simply finding a closed connected graph on the relatively few edges associated with a near-optimal solution (the basic condition for feasibility) may be the most difficult purely combinatorial aspect of the complete problem. With the bound present, the solver appears to be searching almost at random for a low-weight combination of edge variables that describes a closed connected graph. The combinatorial space of numerous edge combinations that are not even feasible seems to be swamping any ability of the solver to progress systematically towards a goal of finding even one closed connected graph on the few edges that are associated with near-optimality. Without the upper bound, the computational prospects are even worse. The solution becomes feasible but now a vast number of graphs are enumerated at far too high an edge count, and an investment of time is made in each of those for routing and capacity considerations. This is not a problem at all for the heuristic, however, as the W1 and S2 steps directly assemble a qualified near-optimal topology for final tuning by J3.

8.5.1 RELAXATIONS FOR LOWER BOUNDS ON OPTIMAL REFERENCE SOLUTIONS

Because of the complexity of the MTRS problem, there was a distinct lack of near-optimal solutions to the full MTRS problem, and so it is difficult to properly assess the absolute solution quality of the heuristic, particularly for larger test case networks. In such cases one generally attempts to see if a tight lower bound on the optimum might be available as a surrogate for optimal reference solutions. A series of simple relaxations were therefore also run, attempting to provide lower bounds to the unrestricted J0 reference solutions. The relaxation strategies were (1) a complete relaxation of all capacity, flow and edge variables, (2) relaxation of only working capacity variables, (3) relaxation of only spare capacity variables, and (4) relaxation of working and spare capacity variables but not edge variables. In all cases restoration and working flow variables were also relaxed (as they were in the previous experiments).

However, none of these strategies yielded useful lower bounds for the J0 problem. Basically, whenever the edge variables are relaxed the solution is fast but it is also meaningless because there is no way to reconcile a fractional edge, and regardless of other relaxations when the edge variables are not relaxed, the problem takes virtually as long to run as the un-relaxed problem. It was already commented above and observed in [39] that for FCR problems, the best bound relaxation produced by the solver is so loose that it is practically meaningless because it corresponds to relaxation of the 1/0 edge variables as well as all flows and capacities. For the complete MTRS problems, we found that the full LP relaxations solved very quickly (in only minutes) but were in the order of 50% of the cost of the true optimal value (when it was available). In fact the full relaxation was always lower that the W1 (FCR) sub-problem of the heuristic alone, clearly demonstrating its lack of utility as a lower bound (and helping to explain why the MIP solver performs so poorly on the full MTRS problem).

In contrast, we found that the capacity-related relaxations are essentially as difficult for the solver as the un-relaxed problem. In nearly all cases, after run times of several hours, the objective values of the J0 relaxations were actually worse than those of the non-relaxed J3 solutions. None of the un-relaxed J0 problems that did not terminate in the allotted time (for which bounds would be the most useful) were able to reach a full termination in capacity-relaxed form either. This is again all consistent with the complexity being dominated by the edge decisions, not the routing and capacity solution.

8.6 CLOSING REMARKS

We have seen from both theoretical and experimental viewpoints that the complexity of the complete problem of topology, routing and spare capacity design for a span-restorable network (MTRS) is very high but that the proposed heuristic produces good solutions very quickly. The heuristic is based on a view of the constituent problems that MTRS contains. It has some aspects that are like a classic FCR problem, which inspires the W1 step. It has other aspects that are like a mesh spare capacity design problem, but where we have to also augment the topology for two-connectedness. This inspires the S2 step. The central hypothesis was that within the set-union of the edges from W1 and S2, a restricted instance of the full problem could find a good solution in far less time than the unrestricted problem. We feel this has been borne out by the results. In the few cases where the full MTRS problem could be solved to near-optimality (the 15-node, 20-node, and 25-node non-full-mesh test cases) the heuristic was within 2.1% of optimal and ran in minutes as opposed to up to hours for the MTRS solution. More typically in the larger test cases, we do not know the actual gap to optimality because the reference solutions could not be solved closely enough to optimality. In these cases, though, we can report that the heuristic produces results in minutes that are as much as 25% better than solutions obtainable with up to 12 hours of runtime on the full MTRS problem.

An unexpected aspect of the attempts at solving the optimal MTRS reference problems was the amount of time the MIP solver would spend before even reaching feasibility in cases where the J3 solution was provided as an upper bound. Related to this was the observation that when J0 tests on full-mesh networks without the bounds were stopped at 12 to 24 hours, their best solution was always associated with many more edges than were optimal. We think this is the key to why MTRS is so hard to solve by MIP methods. Combinatoric principles would have it that there are vastly more arrangements of closed connected graphs with many edges than there are graphs that have relatively few edges that also describe a closed connected graph. But with the LP relaxation being so loose, the MIP progress could be roughly thought of as an almost random walk through the edge-vector space. If it was random, the probability that any node the MIP solver branches to is an even plausible solution graph is extremely low. While this is a simplification, it seems to describe the solver's inability to escape the combinatoric dominance of the overly-connected graphs in the topology space to even find one instance of a low weight closed connected edge vector. The solver is wading around in a combinatoric space in which lightweight closed and connected graphs are extremely rare. Any guidance effect the solver is getting from the loose relaxation of edge variables is swamped by the combinatoric dominance of highly connected graphs.

This also suggests a view of MTRS as a problem that straddles two problem domains. Usually, in ILP optimization problems, there is a vast space of feasible solutions and the problem is to find one that minimizes some cost objective. But another domain of problems are feasibility problems where it is the existence or discovery of a feasible solution that is the challenge. The latter kind of problem is more the purview of constraint programming [80]. We offer a view of MTRS as containing a significant feasibility problem in the construction of low-weight edge vectors that describe closed connected graphs that are even qualified and plausible as transport network graphs, coupled with an optimization problem for routing and capacity. A direction for further research might therefore lie in combining a constraint programming approach for the topology aspect, with an ILP for the routing and capacity aspects.

8.6.1 FUTURE DIRECTIONS

In Section 8.2.2.3, we discussed how an optimal MTRS solution differs quite considerably in terms of $\bar{d}$ and actual cost from lowest-cost benchmark reference solution (i.e., the total cost curves in Figure 8.8 through Figure 8.11). We speculated that this was due to the manner in which the reference solution topologies are essentially a pseudo-random walk through the vast topology space, and so there would be little expectation that a reference solution would approximate the true MTRS optimal solution. An interesting exercise for future work would be to add a constraint to the MTRS ILP model that strictly forces a topology to contain a specified number of spans. Generating MTRS solutions of the 15n30s1 master network with $\Omega = 100$, for instance, forced to have exactly 29 spans, 28 spans, and so on should produce a curve that falls significantly below the red curve in Figure 8.8 (and would also contain the corresponding MTRS solution from Table 8.2). Examination of the resulting optimal topologies might provide added insights into the properties and nature of efficient topologies, and might suggest strategies a planner could use in developing or expanding a network. Analysis of the difference between the benchmark curves and the optimal curve could also yield insightful information, particularly if the two curves have very different behaviours or if the locations of their minima (in terms of $\bar{d}$) show interesting trends.

Future work in this area could more fully study the effects of early termination of W1 and S2 and determine limitations on the approach. Findings also suggest that increasing the edge costs in the W1 and S2 steps might in some cases provide an artificially increased edge set to the J3 step, enough that a slightly improved solution might be possible without significantly hindering run-time. A related improvement might be to develop span-addition algorithms to identify edges not selected by W1 and S2, but which are likely to contribute to a good solution if we include them in the edge set passed to J3. Further work could more fully characterize these effects.

Furthermore, the MTRS problem and three-step heuristic could be applied to path-restorable networks, *p*-cycle networks, SBPP networks, and modular variants. It would be interesting to see if application to these other restoration and protection methods will provide additional insights not seen herein.

We also gain an appreciation of the combinatoric principles of network topology design. There are vastly more arrangements of closed connected graphs with many edges than there are with few edges, and this is the basis for the difficulty the complete MTRS problem has in finding good solutions. M.Sc. work by Chioma Nebo [89] investigated usage of a *graph sifter* that could be used to find feasible (i.e., low-cost bi-connected) topologies. A graph sifter approach might be useful in quickly providing a set of candidate edges and working routing as a replacement to the W1 step, which suffers from the same topology-combinatoric problems as the full MTRS model. The use of properly configured LP-relaxations of W1 and/or S2 (or maybe even the full MTRS problem) could also prove to be an efficient means of quickly identifying a good set of candidate edges for J3.

CHAPTER 9

NODE-INCLUSIVE SPAN RESTORATION[16]

As discussed in CHAPTER 4, two of the main classes of shared-mesh survivability are span-oriented schemes, where restoration or protection paths are established between end-nodes local to the failure, and path-oriented schemes, where restoration paths are established end-to-end. Span-oriented survivability tends to have an inherent speed advantage because its activation process and region of action are more localized to the failure, and also provides better dual failure availability, relative to end-to-end path-oriented schemes [16]. The local action of span restoration also translates into easier control of optical path transmission effects because restoration paths are generally shorter and contained to the region near the failure. In path restoration or SBPP, however, a single span failure can involve many node-pairs all acting simultaneously as failure end-nodes (one node pair per affected lightpath), and their real-time signalling, spare capacity contention, and seizure processing loads are all simultaneously imposed on the network, leading to greater average delay. And because path restoration is performed end-to-end, restoration route lengths tend to be significantly longer than those in span restoration, thereby placing the burden on an even greater number of network nodes.

On the other hand, as long as fully disjoint restoration paths are available, end-to-end restoration has an inherent ability to recover from node failures (for flows transiting the failed node), while span restoration mechanisms do not. The view has long been that at the transport level, failures affecting some part of the outside plant (i.e., spans) are quite frequent (say, weekly) relative to node failures, which are characteristically major and very infrequent events. In addition, line-related outside plant items such as cables, ducts, amplifier housings, etc., are non-redundant elements and so require network level protection, while nodal elements such as an OXC or DCS are, within themselves, already redundant and protected with measures such as 1:N protection on ports and complete 1+1 duplication of switch cores, etc. Span restoration is seen as an effective means of dealing with line and cable-related failures, while the far more rare node failure scenarios are dealt with through high-level service reprovisioning or semi-automated recovery processes such as *iterated capacity scavenging* [47]. Nonetheless, it is of obvious interest if there was some technique that retained the simplicity and localized nature of span restoration while also providing protection for transiting flows through failed nodes. This motivates us to consider a relatively simple but novel extension to basic span restoration that will also support protection of transiting flows upon a node failure.

---

[16] This chapter contains some material previously published in [26].

9.1 BASIC CONCEPTS

In the discussion on span restoration in Section 4.4.1, we defined the end-nodes of the failed span as being the custodial nodes since it is these nodes that act to initiate restoration and anchor the restoration paths that are formed around the failure. Likewise, in a path-restorable network, it is the end-nodes of each failed lightpath that act as custodial nodes (for that lightpath only) since restoration paths are constructed end-to-end rather than between the end-nodes of the failed span. So for the failure indicated in Figure 9.1(a), nodes A and Z are the custodial nodes for the span restoration process, while nodes E and W are the custodial nodes for the path restoration process reacting to the same failure, as shown in Figure 9.1(b).

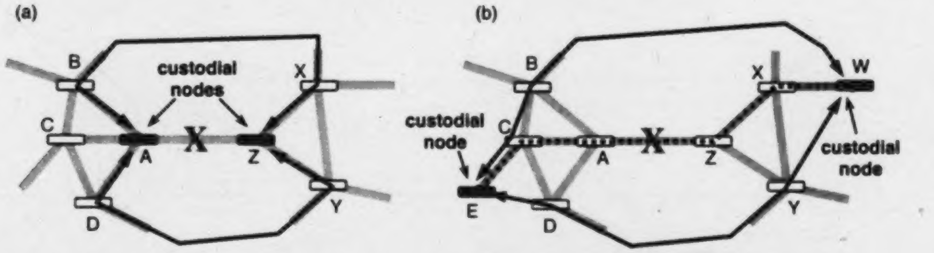


Figure 9.1 – Custodial nodes in (a) span restoration, and (b) path restoration.

We now propose *node-inclusive span restoration* (NISR), which involves taking a specific but highly limited step from span restoration in the continuum of possible options between restoration between the end-nodes of a failed span and end-to-end path restoration. The key idea is that we perform restoration between custodial nodes that are one hop removed from the end-nodes of a failed span along the path of the original working route of the affected demand. In other words, rather than leaving the entire lightpath intact on either side of the failure as in span restoration, or completely rerouting it as in path restoration, the central portion comprising one full hop on either side of the failed span is released and rerouted, while the ends remain intact. Essentially, this amounts to a compromise between span restoration and path restoration in the sense that in NISR, the custodial nodes are further out than they would be in span restoration but not so far out that they reach the end-nodes of the demand as in path restoration.

We illustrate NISR in Figure 9.2. When span A-Z fails in Figure 9.2(a), rather than nodes A and Z acting as custodial nodes as in span restoration, the custodial nodes shift back one hop from the end-nodes of the failure (nodes A and Z) towards the end-nodes of the lightpath. In this case, nodes C and X act as custodial nodes, and two possible restoration routes are as shown. Once this is done, the definition of the protected span entity can be expanded to include the custodial nodes themselves.

In Figure 9.2(b), it is nodes B and Y that act as custodial nodes for the lightpath between nodes F and V since those are the nodes one hop removed from the failure along the route of the lightpath. So unlike span restoration, the custodial nodes for any given failure in NISR are specific to the lightpath, like they are in path restoration, and so there could be multiple custodial node pairs for any given span failure, depending on the number of distinct lightpaths crossing the span. We can therefore think of all the nodes that could possibly act as one of the custodial nodes for lightpaths crossing the failed span as the *custodial regions*. In other words, the set of all nodes one hop away from the custodial nodes define two custodial regions defined with respect to each span failure. For failure of span A-Z in Figure 9.2(c), any lightpath crossing the span will have one custodial node from the group on the left and one from the group on the right. In this particular case, there could be as many as six (2×3) different custodial node pairs simultaneously acting as failure end-nodes seeking restoration routes (in span restoration, there would be only a single pair).

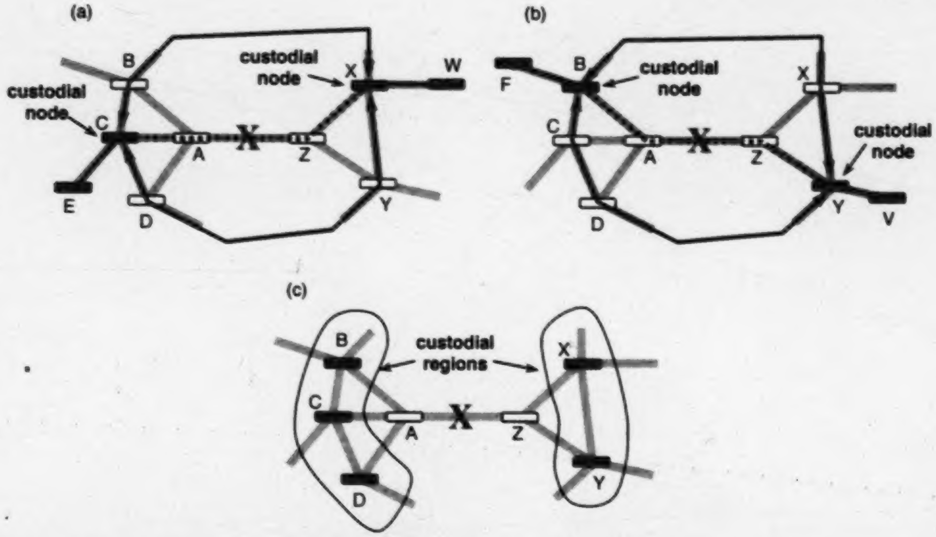


Figure 9.2 – Illustrating node-inclusive span restoration for span failures.

In the general case, there could be substantially more than just six possible custodial node pairs, and in the event that the end-nodes of the failed span are each connected to every other node in the network, there could be O(N^2) potential custodial node pairs, just as there are in path restoration. However, on average, there would be O($\bar{d}^2$) potential custodial node pairs for any span failure. More precisely, there are only $d_A \times d_Z$ potential custodial node pairs, where d_A and d_Z are the nodal degrees of nodes A and Z, respectively[17]. If we can expect nodes to never be more than, say, degree-7 in even the most richly connected networks, there would be a maximum of $(7-1)\times(7-1)=36$ potential custodial node pairs for any failure, which is significantly fewer than the $N\cdot(N-1)/2$ potential pairs for path restoration. In the 15n30s1 network, for instance, the average span failure will involve $\left(\bar{d}-1\right)^2=9$ custodial node pairs for NISR, while a failure in the same network could potentially involve 105 custodial node pairs for path restoration. Solving an NISR rerouting problem (and the associated capacity planning problem) is, therefore analogous to solving a version of the full path restoration problem that involves only a very small subset of all node pairs and is quite localized by comparison.

---

[17] Here, we include the possibility that nodes A and/or Z themselves would act as custodial nodes for lightpaths they terminate. This will be addressed later.

The NISR mechanism will respond identically if a node has failed. The key to this is in understanding how such a failure is perceived by nodes neighbouring the failure. In Figure 9.3(a), when node M fails, the immediate observation from node A will be that there is something amiss with span A-M, and likewise, node Z will deem span M-Z as having failed. So with respect to the lightpath between E and W, if nodes A and Z each instruct the node one hop away along the lightpath (in this case, nodes C and X, respectively) to act as custodial nodes, then restoration routes can be constructed between those nodes and the node failure can be restored. Likewise in Figure 9.3(b), nodes A and Z instruct nodes B and Y, respectively, to act as custodial nodes for the restoration of the lightpath between nodes F and V in the event of the failure of node M.

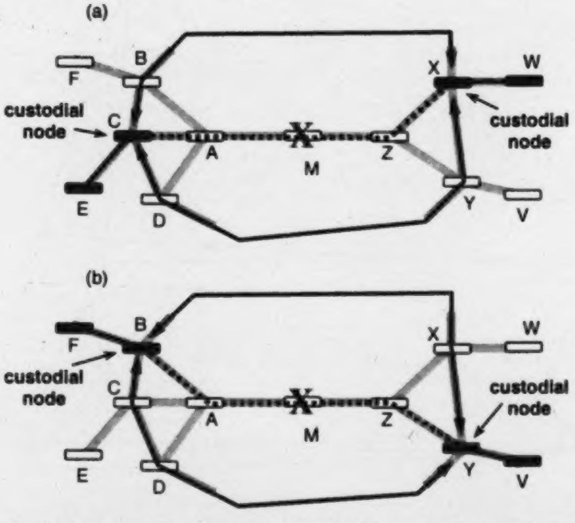


Figure 9.3 – Illustrating node-inclusive span restoration for node failures.

Just as in the event of a span failure, the custodial nodes are specific to the affected demand, and so for node failures too, we can identify custodial regions or subsets of nodes that could potentially be involved in the restoration of a specified node failure. We note, however, that the custodial regions for node failures can be significantly larger than for span failures since node M could easily be of degree-3 or higher. Assuming the failed node and all of its immediate neighbour nodes are all of degree-$\bar{d}$, there would be as many as $(\bar{d}-1)^2 \cdot {}^{\bar{d}}C_2 = (\bar{d}-1)^2 \cdot \bar{d} \cdot (\bar{d}-1)/2 = \bar{d} \cdot (\bar{d}-1)^3/2$ custodial node pairs simultaneously attempting to initiate a restoration process. While this may seem quite high (and it is compared to the number for span failures), it still only amounts to a maximum of 54 custodial node pairs if $\bar{d}=4$. On the other hand, path restoration could have 105 custodial node pairs for each node failure in even a small 15-node network, and as many as 780 for each node failure in a 40-node network.

NISR thus consists of taking the smallest step that leads towards a node-recovery capability, while giving up the least possible in terms of retaining locality of action. It can be either pre-planned or dynamic (and adaptive), and network spare capacity requirements to support this form of span-failure survivability can only improve relative to conventional span restoration capacity design. This follows because the conventional restoration routing solutions always exist as a subset of those that are within the scope of NISR and clearly there will be circumstances where certain loopbacks in the conventional restoration routing solution are eliminated by taking the once-removed standpoint on the rerouting problem.

9.1.1 OPERATIONAL CONSIDERATIONS AND OTHER DETAILS

While span restoration poses only a single commodity maximum-flow type of problem, NISR and path restoration both present simultaneous multi-commodity maximum-flow types of rerouting problems. However, because nodes in a transport network typically have a small number of immediate neighbours, the complexity of the path restoration-like "mini-problems" that arise in NISR are quite strongly bounded in size relative to full-blown path restoration or SBPP problems. In practice, any instance of NISR is unlikely to significantly exceed $\bar{d}^2$ simultaneous commodities for restoration of a span failure, or $\bar{d} \cdot (\bar{d}-1)^3/2$ simultaneous commodities for restoration of a node failure. This not only simplifies and localizes the problem for NISR protection preplanning, but by putting an absolute cap on the complexity it may also permit an efficient distributed and adaptive path restoration protocol (such as the one in [62]) to be employed with extremely high certainty about its worst-case real-time performance.

In Figure 9.4, we consider NISR from the standpoint of a single affected lightpath to gain further insight about its operation. In Figure 9.4(a), a working lightpath is brought down by failure of span B-C. Under NISR, the failure is allowed to propagate part of the way along the lightpath so that the nodes one hop removed from the failure, in this case, nodes A and D, are alerted. This can also be effected by nodes B and C recognizing signal loss and inserting AIS in the surviving path stubs, along with a hop count and node name. In this scenario, nodes A and D are members of custodial regions for failure of span B-C and the restoration path for the particular lightpath shown will be routed between nodes A and D. As discussed earlier, the restoration routes of a given lightpath will vary according to the failure scenario. If span C-D fails, as in Figure 9.4(b), the custodial regions shift and the same lightpath will be restored between nodes B and E.

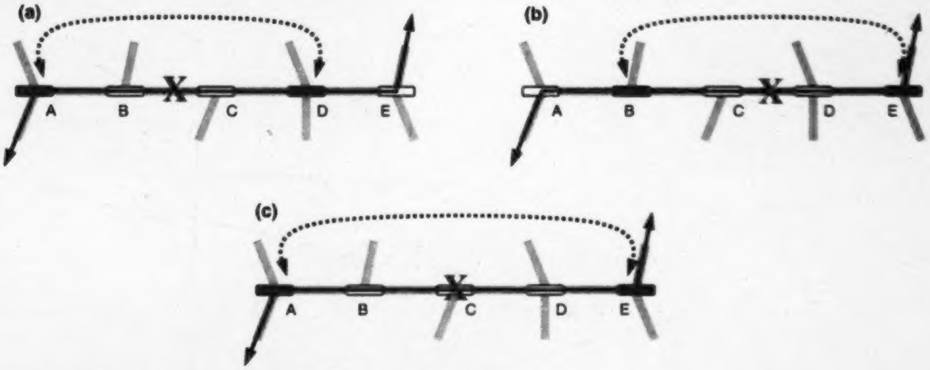


Figure 9.4 – How NISR responds to specific failures.

Finally, in Figure 9.4(c), we consider failure of node C. Since the failure of a node is indistinguishable from the simultaneous failure of its incident spans, then node D detects signal loss and inserts AIS to fail the lightpath back one hop to node E, which becomes a custodial node, and node B does the same with respect to node A, which becomes the other custodial node. Without yet knowing who its corresponding custodial node is, node A can initiate a path restoration process (say, using a protocol like that in [62]) between itself and some other node who also identifies itself as being a custodial node for that lightpath. Node E does likewise. As a result, the initially simple diagrams of Figure 9.4(a)-(b) become somewhat adjusted in a sense. Although all the same concepts apply because of the fundamental inability to immediately distinguish a node failure from failure of its incident spans, the custodial region is effectively referred out *two* hops in the event of a node failure. There is no loss of node failure recovery levels as it may seem, however, because nodes such as A and E in Figure 9.4(c) will still recognize the availability of spans A-B and D-E, respectively, for returning rerouted transit flows to nodes D and B, and if capacity on those spans is needed from a global efficiency standpoint, it will be used. But the advantage gained is that if it is more efficient to not use spans A-B and/or D-E, then referring the restoration process out one additional hop will remove the need for loopback capacity on those spans. And the key to NISR's ability to restore either a span failure or a node failure is that the custodial nodes are independently identified by the two nodes detecting signal loss on either side of the failure, and neither needs to know the identity of the other.

So far, all of the figures we've discussed may appear to illustrate situations where only a single lightpath is crossing a failed span. If we take a closer look at NISR in Figure 9.5, we can observe what happens in a situation where multiple lightpaths with different origins and/or destinations are affected by a single span failure. Here, nodes O_1 and D_1, and nodes O_2 and D_2 each exchange traffic along lightpaths that cross span A-Z. Upon failure of that span, nodes B and Y are identified as the custodial nodes for the blue lightpath between nodes O_1 and D_1 since they are the nodes one hop removed from the nodes that detect the failures (nodes A and Z). The surviving portion of that lightpath between nodes B and Y is then released and the capacity on spans A-B and Y-Z supporting it are made available for restoration routing. At the same time, nodes C and X are identified as being the custodial nodes for the green lightpath between nodes O_2 and D_2, and the surviving portion of that lightpath between nodes C and X is released and the capacity supporting it is made available for restoration routing. The custodial node pairs then simultaneously perform restoration by establishing restoration routes between them via a protocol like the one in [62], and substituting the restoration paths for the failed portions of the working paths.

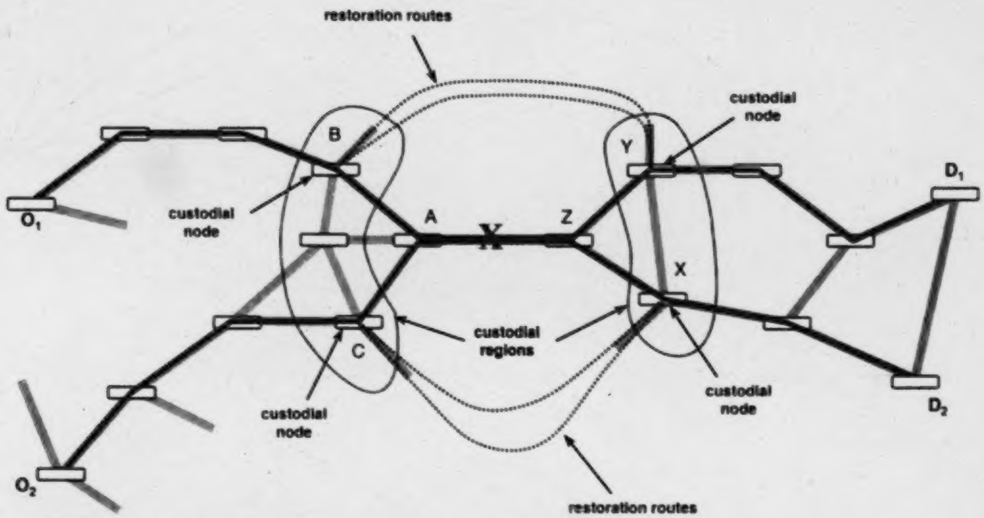


Figure 9.5 – NISR restoring multiple lightpaths simultaneously.

One special case that needs addressing is when an end-node of the failed span is also the end-node of a lightpath crossing it, as shown in Figure 9.6(a). Here, node C detects loss of signal and designates node D as a custodial node for restoration of the lightpath shown. At the same time, node B detects loss of signal and upon inspection determines that it is itself the origin/terminus of the affected lightpath, so it designates itself as the other custodial node. The restoration process is then carried out between nodes B and D. If node C was also an end-node of the affected lightpath, then nodes B and C could each designate themselves as custodial nodes. Interestingly enough, in that situation, then with respect to that particular lightpath, NISR would actually be equivalent to both span restoration (since the custodial nodes are the end-nodes of the failed span) as well as path restoration (since the custodial nodes are also the end-nodes of the lightpath).

A similar scenario is the node failure shown in Figure 9.6(b), where node C fails and one of the adjacent nodes (B) is an end-node of an affected lightpath. Like the previous example, node B detects loss of signal and, realizing that there is no downstream node on the lightpath, it designates itself as a custodial node for the lightpath. Node D also detects the failure and designates node E as the other custodial node, so the restoration process is carried out between nodes B and E, as shown.

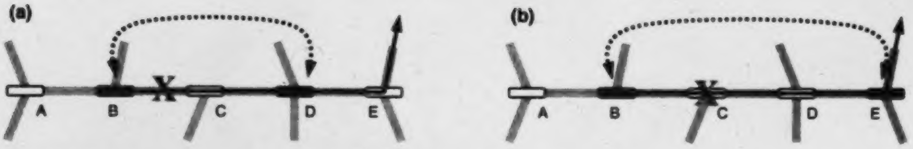


Figure 9.6 – Special cases of NISR restoration.

9.2 NISR DESIGN MODELS

Because NISR extends the restoration process further back along the failed lightpath than span restoration, it promises to be at least as capacity efficient as span restoration. However, since restoration does not generally extend all the way back to the O-D end-nodes of each lightpath, restoration paths are shorter and fewer than in path restoration, and so NISR will not be as capacity-efficient as path restoration. It will be of interest, therefore, to see how NISR compares quantitatively to those survivability mechanisms (as well as to SBPP and *p*-cycle restoration) in capacity efficiency. Like the design models in CHAPTER 5, we develop NISR network capacity design problems as arc-path ILP models.

9.2.1 SPAN-FAILURE RESTORATION

We first address the problem of optimally determining working and restoration routing and placing working and spare capacity so as to allow NISR to provide 100% restoration in the event of any single *span* failure, and call this the NISR-S problem. We begin with knowledge of the graph topology, and all appropriate input parameters and sets (eligible route enumeration, etc.) are pre-processed. To properly represent the model, we introduce new notation as follows:

New Sets:

- N is the set of all possible node pairs, and is usually indexed by *m*.
- $D_i \subseteq D$ is the set demand relations that has at least one eligible working route that crosses span *i*, and would therefore be affected by its failure, and is indexed by *r*.
- $Q_i^r \subseteq Q^r$ is the set of working routes for demand relation *r* that cross span *i*, and would therefore be affected by its failure, and is indexed by *q*.

- P_m is the set of eligible restoration routes between node pair *m*.
- $S_p \subseteq S$ is the set of spans in route *p*.
- $P_{i,m}^{r,q} \subseteq P_m$ is the set of routes between custodial node pair *m* that are eligible for assignment of restoration flow to restore demand relation *r* on working route *q* brought down by failure of span *i*, and is typically indexed by *p*. In other words, $\forall p \in P_{i,m}^{r,q}$ is equivalent to $\forall p \in P_m \big| i \notin S_p$.
- $S_i^{r,q} \subseteq S$ is the set of spans crossed by working route *q* used by demand relation *r* that are between the custodial nodes used to restore a lightpath on that route in the event of failure of span *i*. In other words, they are the spans crossed by the portion of the lightpath that is released upon propagation of the failure indication to the corresponding custodial nodes.

New Input Parameters:

- $N_i^{r,q} \in \mathrm{N}$ is the single NISR custodial node pair to be used by a lightpath routed over working route *q* for demand relation *r* in the event of the failure of span *i*.
- $\delta_{i,m,j}^{r,q,p} \in \{0,1\}$ is a pre-processed input parameter that encodes which spans are crossed by a restoration route. $\delta_{i,m,j}^{r,q,p} = 1$ if restoration route *p* crosses span *j*, and $\delta_{i,m,j}^{r,q,p} = 0$ if it does not (where route *p* is from the set $P_{i,m}^{r,q} \subseteq P_m$).

New Decision Variables:

- $f_{i,m}^{r,q,p} \geq 0$ is the number of restoration paths assigned to eligible route $p \in P_{i,m}^{r,q}$ for failure of span *i*.
- $T_{i,j} \geq 0$ is the total number of channels on span *j* released upon failure of span *i*. These are the working channels on the surviving portions of failed lightpaths that are released upon propagation of the failure indication to the corresponding custodial nodes. The release occurs only on the failed path

segments between nodes of the custodial regions only, with respect to the given failure.

Definition of custodial node pair $N_i^{r,q} \in \mathrm{N}$ is determined by pre-processing for working path q for demand relation r for each failure scenario. The eligible restoration routes between each custodial node pair are enumerated and their routes are encoded in the $\delta_{i,m,j}^{r,q,p}$ parameters by pre-processing as well. In addition to the new notation above, all of the previous notation from earlier models remains. The NISR-S formulation itself is expressed as follows.

$$\text{Minimize} \quad \sum_{\forall j \in S} c_j \cdot \left(s_j + w_j \right) \tag{9.1}$$

$$\text{Subject to:} \quad \sum_{\forall q \in Q^r} g^{r,q} = d^r \qquad \forall r \in D \tag{9.2}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r} \zeta_j^{r,q} \cdot g^{r,q} = w_j \qquad \forall j \in S \tag{9.3}$$

$$\sum_{\forall p \in P_{i,m}^{r,q}} f_{i,m}^{r,q,p} = g^{r,q} \qquad \begin{matrix} \forall i \in S & \forall r \in D_i \\ \forall q \in Q_i^r & \forall m \in \mathrm{N} \middle| m = N_i^{r,q} \end{matrix} \tag{9.4}$$

$$s_j + T_{i,j} \geq \sum_{\forall r \in D_i} \sum_{\forall q \in Q_i^r} \sum_{\forall m \in \mathrm{N} | m = N_i^{r,q}} \sum_{\forall p \in P_{i,m}^{r,q}} f_{i,m}^{r,q,p} \cdot \delta_{i,m,j}^{r,q,p} \qquad \forall i \in S \quad \forall j \in S \middle| i \neq j \tag{9.5}$$

$$T_{i,j} = \sum_{\forall r \in D_i} \sum_{\forall q \in Q_i^r} \sum_{\forall k \in S_i^{r,q} | k = j} g^{r,q} \qquad \forall i \in S \quad \forall j \in S \middle| i \neq j \tag{9.6}$$

The objective function in equation (9.1) is the same as the one in the span-restorable JCA design model, where we seek to minimize the total cost of working and spare capacity required in the network. The constraint sets in equations (9.2) and (9.3) are also carried over from the span-restorable JCA design problem. Equation (9.2) ensures that all working demands are routed, and equation (9.3) places enough working capacity on each span to accommodate that routing. The constraints in equation (9.4) correspond to those in equation (5.2) of the span-restorable SCA model, and ensure that the total restoration flow assigned to all eligible restoration routes between the appropriate custodial nodes is sufficient to fully restore all of the lightpaths on working routes affected by the failure of span i. We note that because the summation is over $\forall p \in P_{i,m}^{r,q}$, which is equivalent to $\forall p \in P_m \middle| i \notin S_p$, then restoration flow will not use routes that cross the failed span. While there is nothing that explicitly forces $f_{i,m}^{r,q,p} = 0$ if $i \in S_p$, a value of $f_{i,m}^{r,q,p} > 0$ will not contribute to the restorability of the failed lightpaths but may require additional spare capacity, so as a consequence, we can disregard the $f_{i,m}^{r,q,p}$ decision variables altogether if $i \in S_p$. Equation (9.5) is equivalent to the spare capacity constraints in equation (5.3) of the span-restorable SCA model, and places enough spare capacity on each surviving span *j* to accommodate the total restoration flow assigned to all restoration routes crossing it for restoration of lightpaths affected by failure of span *i*. The number of working channels, $T_{i,j}$, released on span *j* in the event of failure of span *i*, is calculated by equation (9.6) and is applied as a credit to the spare capacity allocated by equation (9.5).

This model describes a JCA problem since working and restoration routing are done simultaneously. If shortest path routing or some other method of routing working demands is done separately, prior to optimization of restoration routing and spare capacity, we can convert the above JCA model into a corresponding SCA model by making a few key changes. First, the w_j variables become unnecessary and can be removed from the objective function, and the constraints in equations (9.2) and (9.3) no longer apply. Unlike the span-restorable SCA model, we still require $g^{r,q}$ values since they are needed in equation (9.4), but it is now an input parameter rather than a decision variable. Finally, the calculation of $T_{i,j}$ in equation (9.6) can be preprocessed since $g^{r,q}$ is no longer a decision variable. We also note that as formulated, working lightpaths for an given demand relation are allowed to be routed over possibly several distinct working routes. If pure shortest path routing (or some method where working demands can be routed on only a single working route) is used, this extra dimension of the model could be removed.

9.2.2 NODE-FAILURE RESTORATION

We now address the problem of optimally determining the working and restoration routing and placing working and spare capacity so as to allow NISR to provide 100% restoration in the event of any single *node* failure, and call this the NISR-N problem. As with the NISR-S problem, we begin with knowledge of the graph topology, and all appropriate input parameters and sets (eligible route enumeration, etc.) are pre-processed. To properly represent the model, we introduce more new notation as follows:

New Sets:

- $\tilde{D}_n \subseteq D$ is the set of demand relations that have at least one eligible working route that crosses node *n*, and would therefore be affected by its failure, and is indexed by *r*.
- $\tilde{Q}_n^r \subseteq Q^r$ is the set of working routes for demand relation *r* that cross node *n*, and would therefore be affected by its failure, and is indexed by *q*.
- $\tilde{S}_n^{r,q} \subseteq S$ is the set of spans crossed by working route *q* used by demand relation *r* that are between the custodial nodes used to restore a lightpath on that route in the event of failure of node *n*. In other words, they are the spans crossed by the portion of the lightpath that is released upon propagation of the failure indication to the corresponding custodial nodes.
- $N_p \subseteq N$ is the set of nodes crossed by route *p*.
- $\tilde{P}_{n,m}^{r,q} \subseteq P_m$ is the set of routes between custodial node pair *m* that are eligible for assignment of restoration flow to restore demand relation *r* on working route *q* brought down by failure of node *n*, and is typically indexed by *p*. In other words, $\forall p \in \tilde{P}_{n,m}^{r,q}$ is equivalent to $\forall p \in P_m \middle| n \notin N_p$.

New Input Parameters:

- $\tilde{N}_n^{r,q} \in \mathrm{N}$ is the single NISR custodial node pair to be used by a lightpath routed over working route *q* for demand relation *r* in the event of the failure of node *n*.
- $\tilde{\delta}_{n,m,j}^{r,q,p} \in \{0,1\}$ is a pre-processed input parameter that encodes which spans are crossed by a restoration route. $\tilde{\delta}_{n,m,j}^{r,q,p} = 1$ if restoration route *p* crosses span *j*, and $\tilde{\delta}_{n,m,j}^{r,q,p} = 0$ if it does not (where route *p* is from the set $\tilde{P}_{n,m}^{r,q} \subseteq P_m$).

New Decision Variables:

- $\tilde{g}_n^{r,q} \geq 0$ is the number of working channels employed on working route *q* for demand relation *r* prior to failure, that do *not* terminate at node *n*.
- $\tilde{T}_{n,j} \geq 0$ is the total number of channels on span *j* released upon failure of node *n*. These include the working channels on the surviving portions of failed lightpaths that are released upon propagation of the failure indication to the corresponding custodial nodes, as well as all of the inherently non-restorable working paths that source/sink at the failed node.
- $\tilde{f}_{n,m}^{r,q,p} \geq 0$ is the number of restoration paths assigned to eligible route $p \in \tilde{P}_{n,m}^{r,q}$ for failure of node *n*.

Definition of custodial node pair $\tilde{N}_n^{r,q} \in \mathbf{N}$ is determined by pre-processing for working path *q* for demand relation *r* for each failure scenario. The eligible restoration routes between each custodial node pair are enumerated and their routes are encoded in the $\tilde{\delta}_{n,m,j}^{r,q,p}$ parameters by pre-processing as well. In addition to the new notation above, all of the previous notation from earlier models remains. The NISR-N formulation itself is expressed as follows.

$$\text{Minimize} \quad \sum_{\forall j \in S} c_j \cdot \left(s_j + w_j\right) \tag{9.7}$$

$$\text{Subject to:} \quad \sum_{\forall q \in Q^r} g^{r,q} = d^r \qquad \forall r \in D \tag{9.8}$$

$$\sum_{\forall r \in D} \sum_{\forall q \in Q^r} \zeta_j^{r,q} \cdot g^{r,q} = w_j \qquad \forall j \in S \tag{9.9}$$

$$\sum_{\forall p \in \tilde{P}_{n,m}^{r,q}} \tilde{f}_{n,m}^{r,q,p} = g^{r,q} \qquad \begin{array}{c} \forall n \in N \quad \forall r \in \tilde{D}_n \\ \forall q \in \tilde{Q}_n^r \,\middle|\, O_r \neq n \neq T_r \\ \forall m \in \mathbf{N} \,\middle|\, m = \tilde{N}_n^{r,q} \end{array} \tag{9.10}$$

$$s_j + \tilde{T}_{n,j} \geq \sum_{\forall r \in \tilde{D}_n} \sum_{\forall q \in \tilde{Q}_n^r} \sum_{\forall m \in \mathbf{N} | m = \tilde{N}_n^{r,q}} \sum_{\forall p \in \tilde{P}_{n,m}^{r,q}} \tilde{f}_{n,m}^{r,q,p} \cdot \tilde{\delta}_{n,m,j}^{r,q,p} \qquad \forall n \in N \quad \forall j \in S \tag{9.11}$$

$$\tilde{T}_{n,j} = \sum_{\forall r \in \tilde{D}_n} \sum_{\forall q \in \tilde{Q}_n^r} \sum_{\forall k \in \tilde{S}_n^{r,q} | k = j} g^{r,q} \qquad \forall n \in N \quad \forall j \in S \tag{9.12}$$

As with previous design models, the objective function in equation (9.7) minimizes the total cost of capacity. The constraints in equations (9.8) and (9.9) are identical to those in equations (9.2) and (9.3) from the NISR-S problem, and ensure that all working demands are routed and that there is enough working capacity placed on each span to accommodate that routing. The constraint sets in equations (9.10), (9.11), and (9.12) are equivalent to those in equations (9.4), (9.5), and (9.6), except that they have been modified to correspond to node failure restoration. Equation (9.10) ensures that the total restoration flow assigned to all eligible restoration routes between the appropriate custodial nodes is sufficient to fully restore all of the lightpaths on working routes affected by the failure of node *n*. Working routes for demand relations that originate or terminate at node *n* are excluded ($(O_r \neq n \neq T_r)$) from consideration since they are inherently unrestorable. And we note that because the summation is over $\forall p \in \tilde{P}_{n,m}^{r,q}$, which is equivalent to $\forall p \in P_m \middle| n \notin N_p$, then restoration flow will not use routes that cross the failed node. While there is nothing that explicitly forces $\tilde{f}_{n,m}^{r,q,p} = 0$ if $n \in N_p$, a value of $\tilde{f}_{n,m}^{r,q,p} > 0$ will not contribute to the restorability of the failed lightpaths but may require additional spare capacity, so as a consequence, we can disregard the $\tilde{f}_{n,m}^{r,q,p}$ decision variables altogether if $n \in N_p$. Equation (9.11) places enough spare capacity on each surviving span *j* to accommodate the total restoration flow assigned to all restoration routes crossing it for restoration of lightpaths affected by failure of node *n*, with credit for $\tilde{T}_{n,j}$, the number of working channels released on span *j* in the event of failure of node *n*. $\tilde{T}_{n,j}$ is calculated by equation (9.12).

As mentioned above, there are no constraints that explicitly force no restoration flow over restoration routes that cross a failed node since this will naturally be the case by way of the interaction of the objective function and equations (9.10) and (9.11). However, we can apply value-added constraints as in equation (9.13), below, to do just that.

$$\sum_{\forall p \in P_m | n \in N_p} \tilde{f}_{n,m}^{r,q,p} = 0 \qquad \begin{array}{c} \forall n \in N \quad \forall r \in \tilde{D}_n \\ \forall q \in \tilde{Q}_n^r \middle| O_r \neq n \neq T_r \\ \forall m \in \mathrm{N} \middle| m = \tilde{N}_n^{r,q} \end{array} \qquad (9.13)$$

We can also add another such constraint to force $\tilde{f}_{n,m}^{r,q,p} = 0$ for all demand relations r where $n = O_r$ or $n = T_r$, or in other words, all working routes for demands that originate or terminate at the failed node will have no restoration at all. The constraints in equation (9.14) will assert this.

$$\sum_{\forall p \in P_m} \tilde{f}_{n,m}^{r,q,p} = 0 \qquad \begin{array}{c} \forall n \in N \quad \forall r \in \tilde{D}_n \\ \forall q \in \tilde{Q}_n^r \middle| ((O_r = n) or (T_r = n)) \\ \forall m \in \mathrm{N} \middle| m = \tilde{N}_n^{r,q} \end{array} \qquad (9.14)$$

As with the NISR-S model, the NISR-N formulation presented above is a JCA design model since the working routing and working capacity allocation is performed in coordination with the restoration routing and spare capacity allocation. If shortest path routing or some other method of routing working demands is done separately, prior to optimization of restoration routing and spare capacity, we can convert the JCA model into a corresponding SCA model by making the same changes as with the NISR-S model. The w_j variables can be removed from the objective function, and the constraints in equations (9.8) and (9.9) no longer apply. We still require $g^{r,q}$ values since they are needed in equation (9.10), but it is now an input parameter rather than a decision variable. The calculation of $\tilde{T}_{n,j}$ in equation (9.12) can be preprocessed since $g^{r,q}$ is no longer a decision variable.

9.2.3 COMBINED SPAN-FAILURE AND NODE-FAILURE RESTORATION

It's quite obvious that a network designed to be fully restorable to every single span failure will not necessarily be fully restorable to every single node failure since there may not be sufficient capacity available for the multiple span failures that would arise as a consequence. However, while it may at first seem counterintuitive, it is also possible that an NISR network is fully restorable to all single node failures but not so to every span failure. We illustrate this in Figure 9.7, below, where a single unit-capacity working lightpath shown in purple is routed on a network, and one unit of spare capacity is available only on the network spans indicated. When node B fails, the NISR mechanism restores the lightpath between custodial nodes A and D, and the portion of the lightpath between nodes A and D is replaced by a restoration route on the spans in blue, each of which has a single unit of spare capacity available. Likewise, if nodes C, D, or E fail, restoration routes crossing the orange, green, and red spans, respectively, will be constructed by the NISR mechanism. Failure of either node A or Node F (end-nodes of the lightpath itself) is inherently unrestorable and so is not considered, and failure of any of the other nodes will not affect the lightpath. It is therefore clear that the network shown is fully restorable so any single node failure. However, if span X fails, it is impossible to construct a restoration route within the available spare capacity indicated in the figure, so the network is not fully restorable to span failures.

We may then wish to design a network to be fully restorable to any single span failure as well as to any single node failure. Such a model can be constructed by simply combining the NISR-S and NISR-N formulations. Adding the node restoration constraints in equations (9.10), (9.11), and (9.12) from the NISR-N model to the NISR-S model will result in a combined model we can denote by NISR-NS.

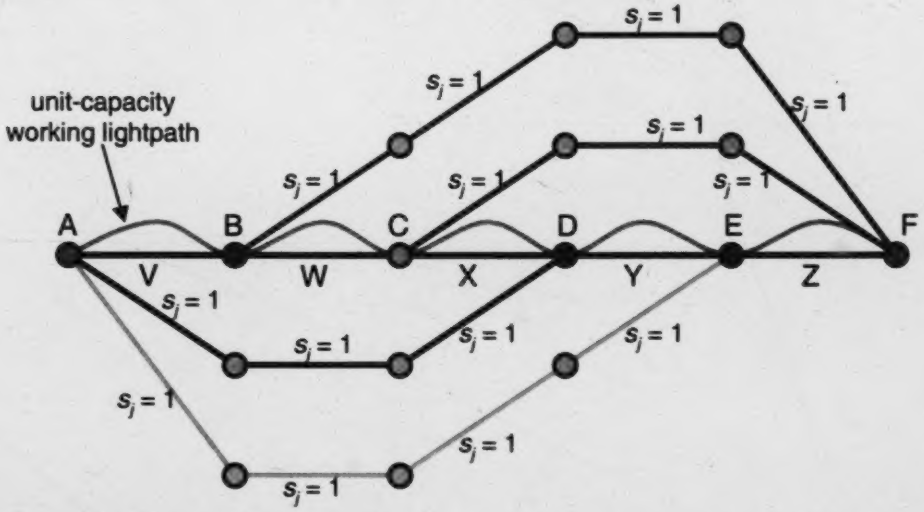


Figure 9.7 – With NISR, 100% node-failure restoration doesn't necessarily imply 100% span-failure restoration.

9.3 EXPERIMENTAL STUDY METHOD

To validate the NISR concept and to characterize the capacity requirements relative to the conventional span-restorable and path-restorable schemes, we conducted comparative capacity design trials using the test-case network topologies described in Section 6.1. As with the benchmark designs in CHAPTER 6, the meta-mesh design model was implemented in AMPL Mathematical Programming Language version 9.0 and solved with CPLEX 9.0 MIP Solver running on a 4-processor SUN UltraSparc III running at 900 MHz with 16 GB of RAM. Pre-processing for eligible working and restoration routes, and other input parameters was done on a dual-processor AMD Opteron 242 PC with 1 GB of RAM, running Windows 2000. All working and spare capacity allocations were integer, corresponding to capacity design and restoration mechanisms at the wavelength level. Results are based on a full CPLEX termination with a MIPGAP setting of 10^{-4} (i.e., solutions are guaranteed to be within 0.01% of optimal). All took less than several minutes to solve, and except for the most richly connected networks from the largest network families, they actually took no more than several seconds. However, because of the manner in which the ILP model was formulated within AMPL, actually generating the data files for use by CPLEX was very time-consuming for the larger networks. In fact, some of the 35n70s1 test case networks required several days just for proper pre-processing to map working route and custodial nodes for all possible failure scenarios. As such, the 40n80s1 networks were not tested in this chapter, leaving 124 test case networks in five families.

For the JCA designs, pre-processing of the eligible working route sets enumerated the 5 shortest routes between each O-D node pair, and in the SCA designs, working routing was via shortest paths. For both the SCA and JCA designs, pre-processing of eligible restoration route sets enumerated the shortest routes between each custodial node pair so that there are at least 10 distinct eligible restoration routes between each custodial node pair for each failure scenario.

9.4 RESULTS AND DISCUSSION

Results of node-inclusive span restoration design experiments are presented and discussed in the coming sections. We first examine NISR network capacity requirements in the same manner as the various benchmark survivability mechanisms were studied in Section 6.4, and then compare them to span-restorable designs since span restoration is the basic mechanism most similar to node-inclusive span restoration.

9.4.1 NISR-S CAPACITY COSTS

Figure 9.8 through Figure 9.12 show normalized spare capacity costs of optimally designed (i.e., minimum cost) networks of the various families designed to be 100% restorable using NISR-S restoration with SCA design, as well as the equivalent designs using the SCA models for span restoration, *p*-cycle restoration, SBPP, and path restoration. Each figure provides data for a single network family, and each data point represents the total wavelength-kilometres of spare capacity required to provide full restoration in the optimal solution for the member of the family with the indicated average nodal degree ($\bar{d}$) using the specified survivability mechanism when working capacity is determined via shortest path routing. In each figure, capacity costs have been normalized to the same lowest-cost design over all networks in the family and all survivability mechanisms as we used in Section 6.4. Within each figure, data points have been organized into curves corresponding to the various survivability mechanisms.

We can observe qualitatively from Figure 9.8 through Figure 9.12 that, like the metamesh network designs of CHAPTER 7, NISR-S network designs require more capacity than the path restoration and SBPP designs and less than the span restoration and *p*-cycle designs. Over all of the 124 test cases studied, NISR-S restoration required an average of 19.7% less spare capacity than span restoration, 17.1% less than *p*-cycle restoration, 12.9% more than SBPP, and 20.5% more than path restoration. It is also quite apparent from the figures that NISR-S was particularly capacity-efficient relative to the other mechanisms in highly connected networks. In test case networks with $\bar{d} \geq 3.0$, NISR-S required 28.0% less spare capacity than span restoration, and 24.9% less than *p*-cycle designs. In those same networks, NISR-S spare capacity costs were only 10.7% more than SBPP and 17.0% more than path restoration. The significant drop in spare capacity in all of the conventional benchmark survivability mechanisms in the vicinity of $\bar{d} \cong 3.0$ that we observed in Section 6.4 is evident in the NISR-S spare capacity cost curves here as well.

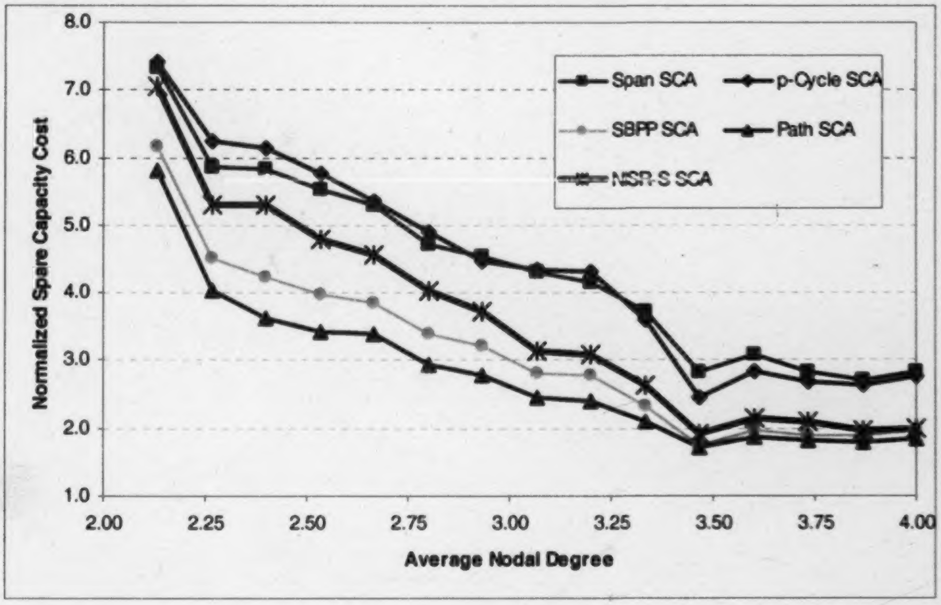


Figure 9.8 – NISR-S SCA normalized spare capacity costs for the 15n30s1 network family.

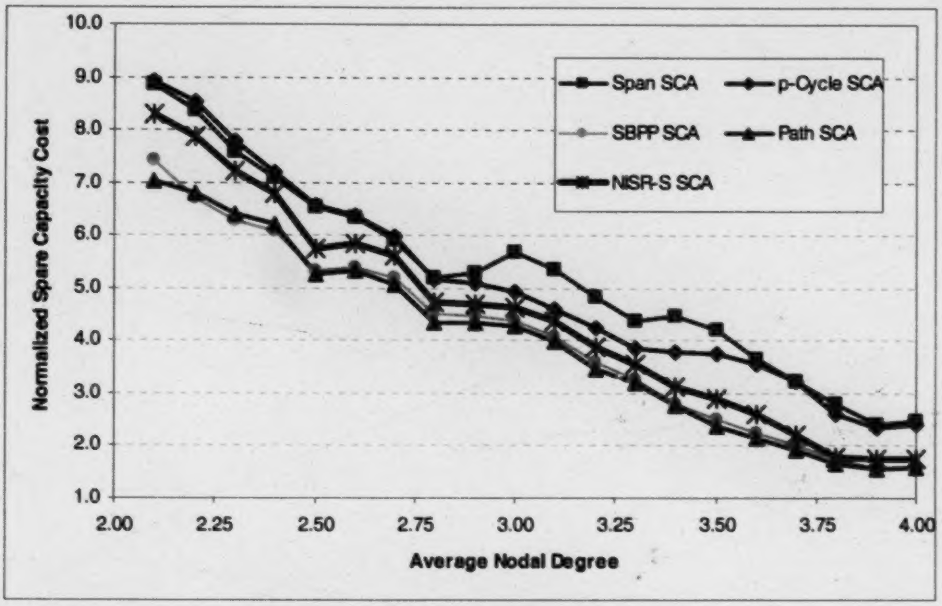


Figure 9.9 – NISR-S SCA normalized spare capacity costs for the 20n40s1 network family.

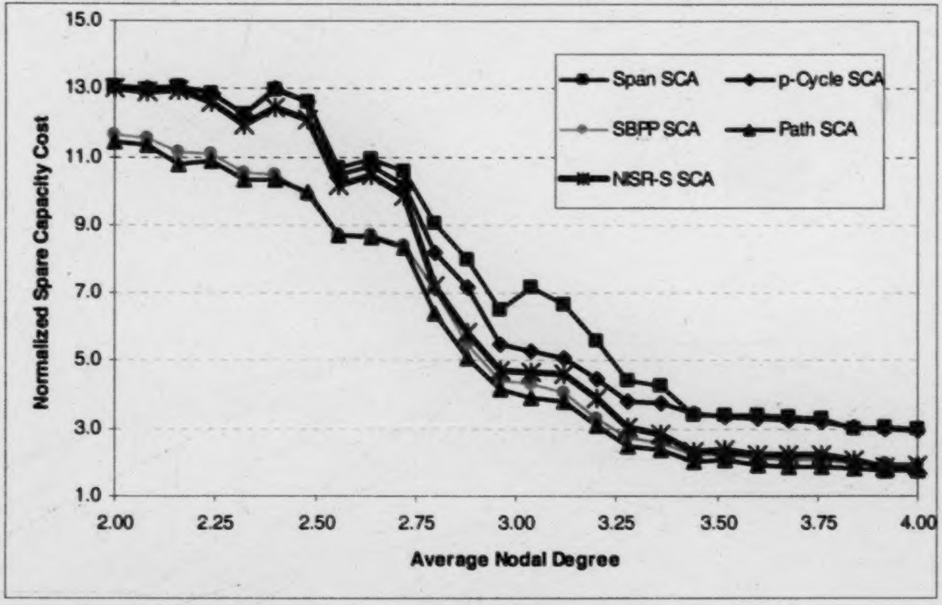


Figure 9.10 – NISR-S SCA normalized spare capacity costs for the 25n50s1 network family.

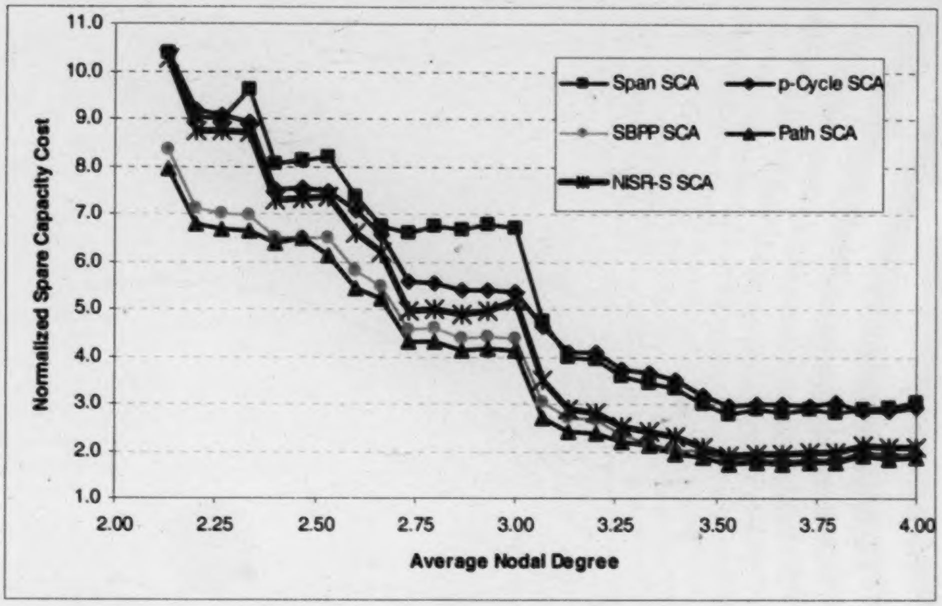


Figure 9.11 – NISR-S SCA normalized spare capacity costs for the 30n60s1 network family.

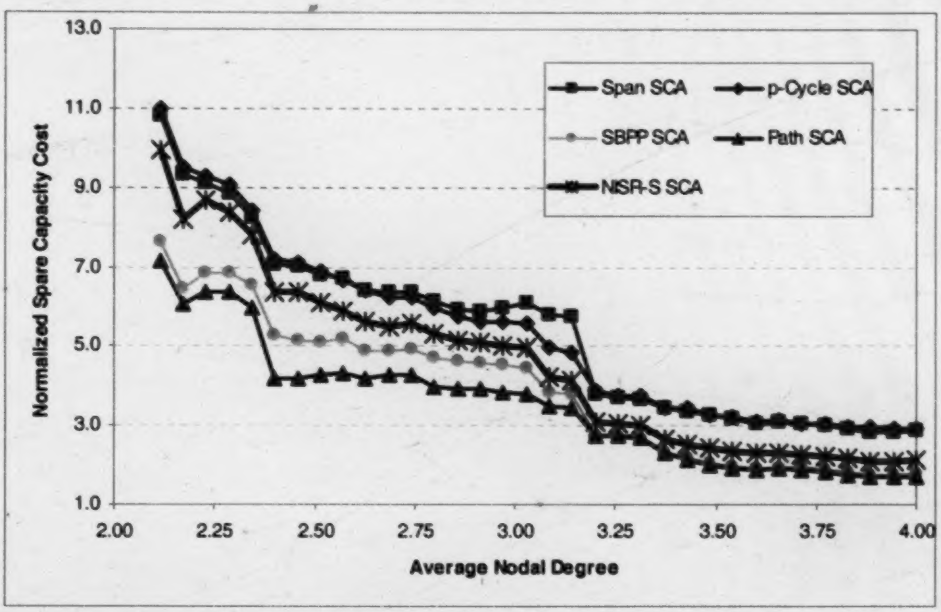


Figure 9.12 – NISR-S SCA normalized spare capacity costs for the 35n70s1 network family.

9.4.1.1 NISR-S GAP RATIO RELATIVE TO SPAN AND PATH RESTORATION

It is also evident from Figure 9.8 through Figure 9.12 that the NISR-S spare capacity curves undertake what we could describe as a state change in the vicinity of $\bar{d} \cong 3.0$. In networks with $\bar{d} \ll 3.0$, the NISR-S curves are quite close to the span restoration and *p*-cycle curves, while in networks with $\bar{d} \geq 3.0$, the NISR-S curves all tend to deviate significantly and approach the SBPP and path restoration curves. This seems to suggest that NISR behaves more like a span-restorable survivability mechanism in sparse networks, and more like end-to-end-restorability mechanisms in richly connected networks.

One possible explanation is that in more highly connected networks, many more working lightpaths are able to follow more-or-less direct routes of only a few hops across the network. The shorter the working route is, the closer NISR will be to true path restoration since a greater proportion of the lightpath's working capacity will be stub-released and the distance between the custodial nodes will be a greater fraction of the total working route length. In fact, when NISR is used to restore a failed span on a working route that is only three hops in length, it will be identical to path restoration (if the failure is the middle span in the route) since the custodial nodes will actually be the end-nodes of the working route itself. For working routes only slightly longer than three hops, the differences between NISR-S and path restoration will not be great. On the other hand, in networks with $\bar{d} \ll 3.0$, there will be a significant amount of working lightpaths routed over much more lengthy working routes, which may even include at least several segments over degree-2 chains. When NISR is used to restore these longer working routes, its behaviour will be closer to that of span restoration than path restoration since the proportional amount of working capacity that is stub-released will be quite small relative to the total working capacity required by the route.

This behaviour is even more evident in Figure 9.13 through Figure 9.17, which plot the *gap ratios* of the NISR-S capacity costs between span restoration and path restoration capacity costs. More precisely, each data point in the figures is equivalent to the ratio of the difference between the NISR-S and path restoration capacity costs for a given test case network and the difference between the span restoration and path restoration capacity costs for that same network. In other words, the NISR-S capacity cost of each test case network has been normalized or scaled so that the span restoration capacity cost and path restoration capacity of that network are equivalent to 1.0 and 0.0, respectively. The gap ratios of the 15n30s1 and 20n40s1 network families appear somewhat linear, and the linear regression trend lines (not shown) for those families have $R^2 = 0.9361$ and $R^2 = 0.8231$, respectively, as well as correlation coefficients of $r_{xy} = 0.968$ and $r_{xy} = 0.907$, respectively, suggesting very strongly linear relationships. However, the gap ratios of the 25n50s1 and 35n70s1 network families (and to a lesser extent, the 30n60s1 family) actually appear to be bimodal with a flat section of high gap ratio (i.e., closer to span restoration) in networks with $\bar{d} \ll 3.0$, a flat section of low gap ratio (i.e., closer to path restoration) in networks with $\bar{d} \gg 3.0$, and a small middle section where the gap ratio drops rapidly with increasing average nodal degree. Interestingly, the location (with respect to $\bar{d}$) of this rapid drop in gap ratio corresponds very closely with the location of the drop and levelling observed in spare capacity cost curves for all of the other survivability mechanisms in Section 6.4. The simple interpretation then, is that the combination of factors that cause the acceleration in spare capacity decreases with increasing $\bar{d}$ at $\bar{d} \cong 3$ in Section 6.4 are even more dominant in NISR-S than for the benchmark survivability mechanisms. This is perhaps due to the increased association that the capacity requirements of NISR-S have (relative to the other survivability mechanisms) with working route lengths or, more precisely, with routes of three hops or shorter. For the moment, we will take a closer look at the nature of the relationship between the gap ratio and $\bar{d}$, and will then revisit possible reasons later towards the end of this section of the thesis.

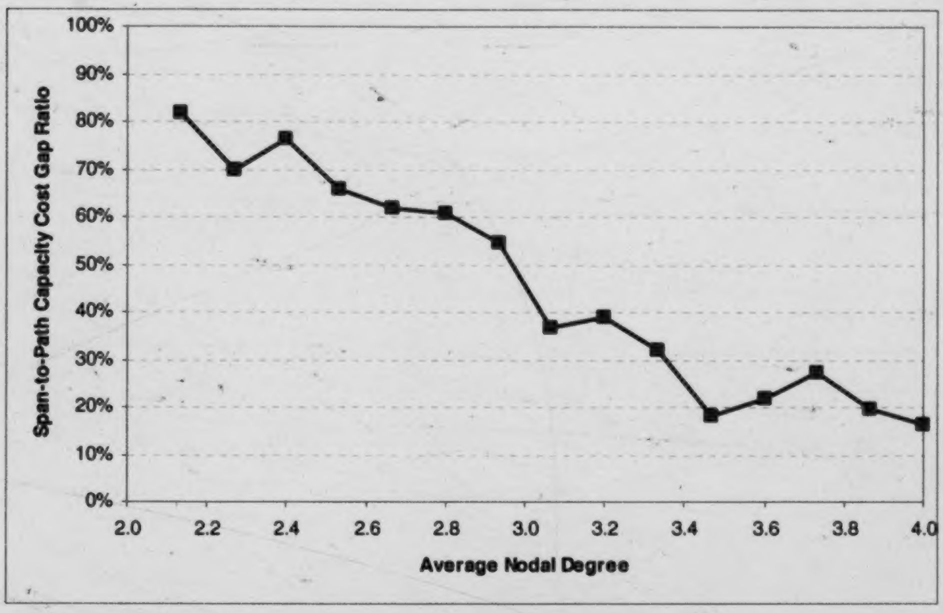


Figure 9.13 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 15n30s1 network family.

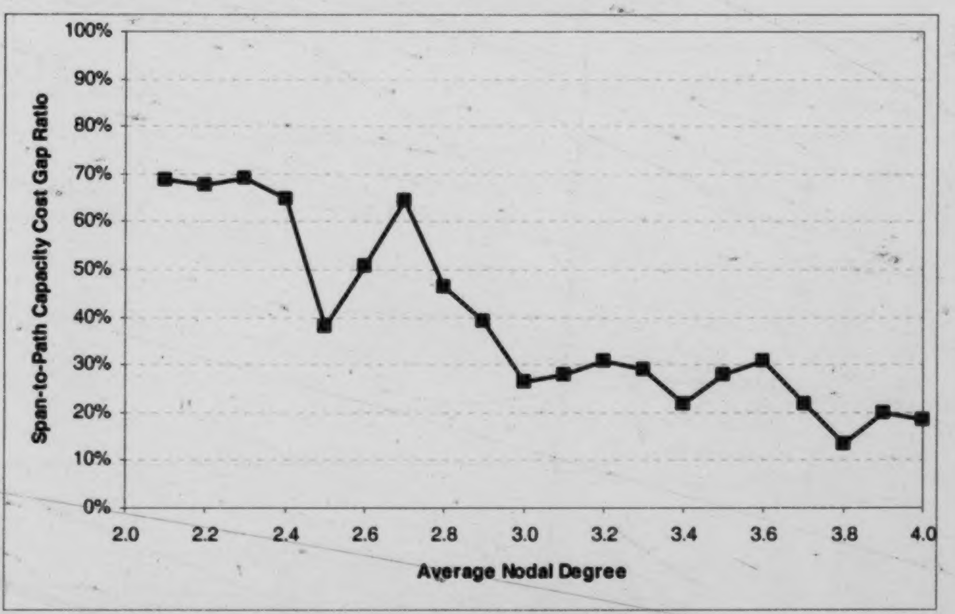


Figure 9.14 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 20n40s1 network family.

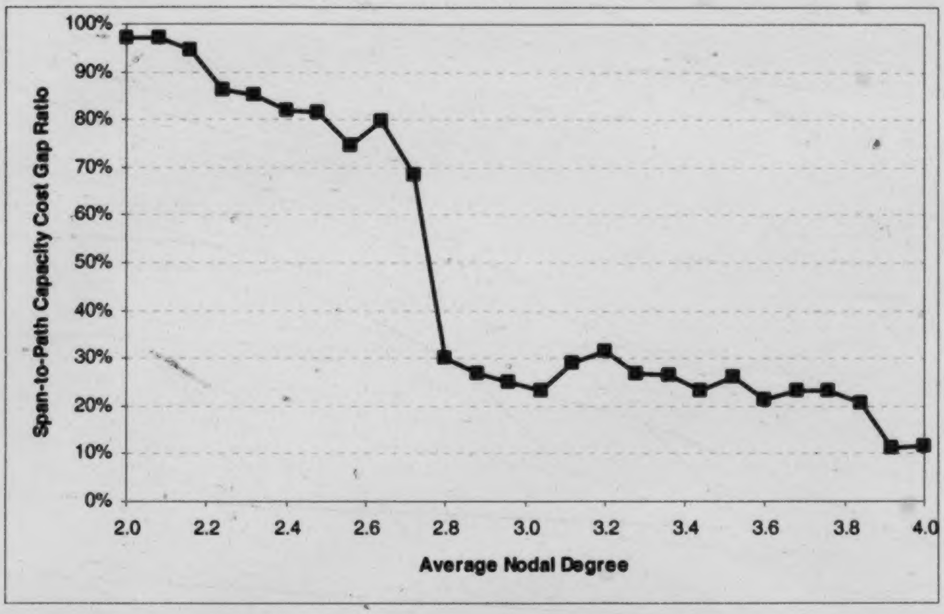


Figure 9.15 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 25n50s1 network family.

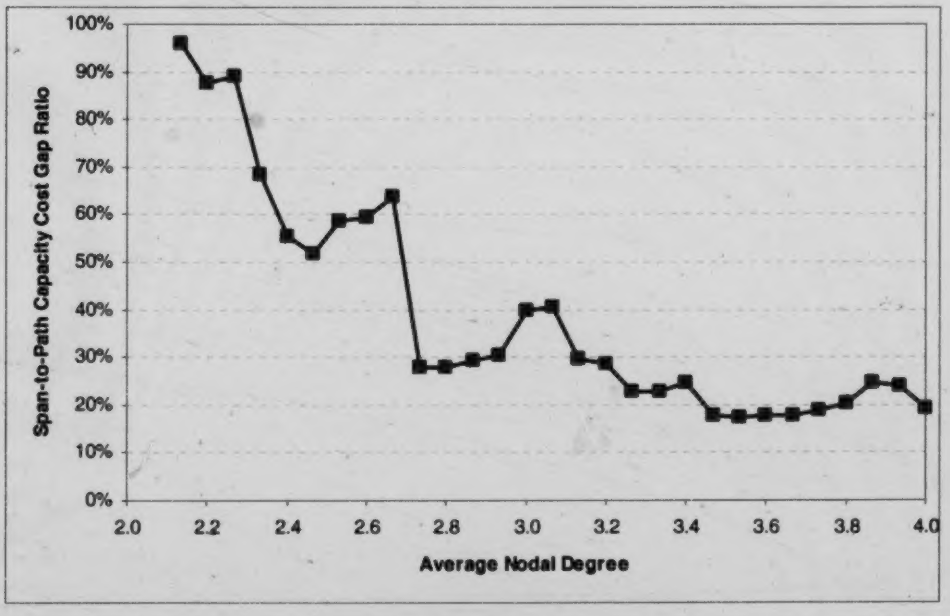


Figure 9.16 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 30n60s1 network family.

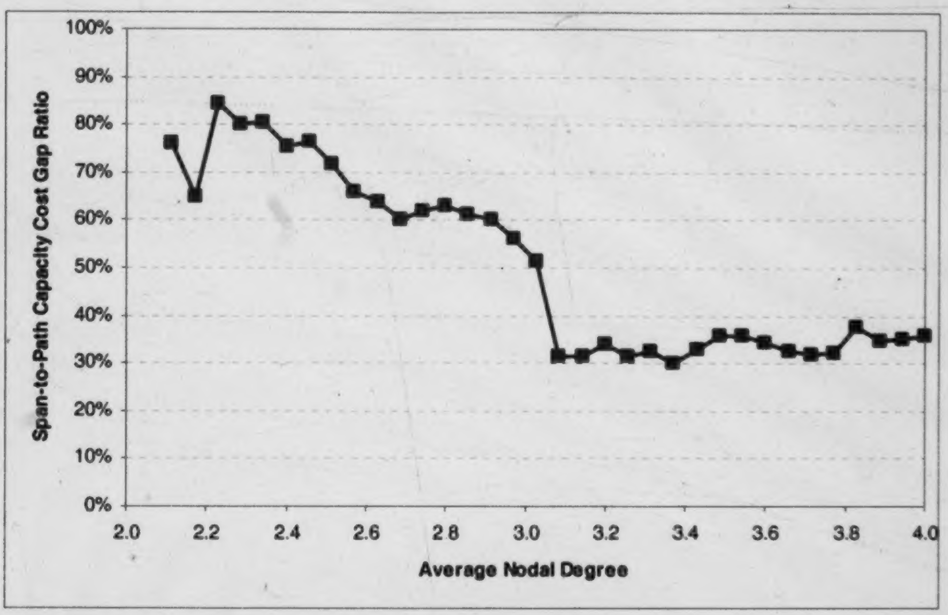


Figure 9.17 – Gap ratio of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for the 35n70s1 network family.

In Figure 9.18, all of the data points in Figure 9.13 through Figure 9.17 are placed in a single scatter plot in an effort to more definitively identify an overall trend in NISR-S gap ratios. The straight linear regression trend line shown in blue appears to be a reasonably good fit, with $R^2 = 0.7594$. However, a *curvilinear* or *polynomial* regression curve [64] shown by the green curve appears to be an even better fit, with an $R^2 = 0.8301$. In this case, the data has been fit to a polynomial curve of the form in equation (9.15).

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 \cdot x + \hat{\beta}_2 \cdot x^2 + \hat{\beta}_3 \cdot x^3 + \hat{\beta}_4 \cdot x^4 \qquad (9.15)$$

The $\hat{\beta}_0$, $\hat{\beta}_1$, $\hat{\beta}_2$, $\hat{\beta}_3$, and $\hat{\beta}_4$ coefficients in the polynomial are estimated by the least squares method to minimize the sum of the squares of the vertical error between the actual data points and the points on the curve [64]. More precisely, the coefficients are determined such that the function in equation (9.16) is minimized, which is accomplished by differentiating with respect to each of the coefficients, equating those partial derivatives with zero, and solving the set of equations that result. More information on the method can be found in [64] and [85].

$$\sum_{\forall i\in\{1...n\}}\left[y_i-\left(\hat{\beta}_0+\hat{\beta}_1\cdot x_i+\hat{\beta}_2\cdot x_i^2+\hat{\beta}_3\cdot x_i^3+\hat{\beta}_4\cdot x_i^4\right)\right]^2 \tag{9.16}$$

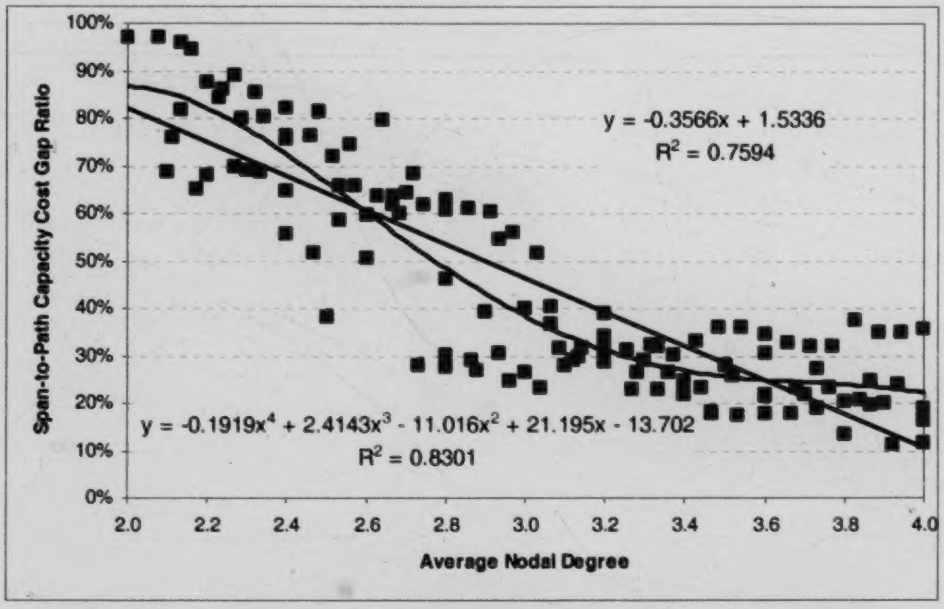


Figure 9.18 – Scatter plot of the gap ratios of NISR-S SCA capacity costs between span restoration SCA and path restoration SCA capacity costs for all 124 test case networks.

We note that the polynomial regression curve of Figure 9.18 appears to fit the general bimodal pattern of a relatively flat section with high gap ratio at low $\bar{d}$, a flat section of low gap ratio at high $\bar{d}$, and a middle section with a relatively rapid decrease as $\bar{d}$ increases. Two other curves not shown were fit to the data as well (one was of the form $\hat{y}=\hat{\beta}_0+\hat{\beta}_1\cdot x+\hat{\beta}_2\cdot x^2$ and the other fit $\hat{y}=\hat{\beta}_0+\hat{\beta}_1\cdot x+\hat{\beta}_2\cdot x^2+\hat{\beta}_3\cdot x^3$). Although they both had $R^2>0.7594$ (better than a linear regression line), neither was as good a fit as the curve in Figure 9.18, nor did they match as closely to the bimodal relationship observed in the gap ratios of the 25n50s1 and 35n70s1 network families. While we cannot suggest that the precise curve that most closely matches the relationship between the NISR-S gap ratio and $\bar{d}$ is the one in Figure 9.18 with the exact equation $y=-13.702+21.195\cdot x-11.016\cdot x^2+2.2143\cdot x^3+-0.1919\cdot x^4$ (or even that it is of the form $\hat{y}=\hat{\beta}_0+\hat{\beta}_1\cdot x+\hat{\beta}_2\cdot x^2+\hat{\beta}_3\cdot x^3+\hat{\beta}_4\cdot x^4$), the basic idea is that the data appears to more closely match a bimodal curve segment than a linear one.

We now return to the earlier discussion regarding the possible dependence of the NISR-S gap ratio to the length of working routes used, or to the number of working routes with a length of three hops or less. It is obvious that when working routing is via shortest paths, the average length of those working routes will necessarily increase as the average nodal degree of the network decreases. At the same time, the more sparsely connected the network is, the greater the proportion of all working routes that will be below a given length or number of hops[18]. So we would expect that a scatter plot of average nodal degree versus the average working route length would show some sort of relationship, and indeed it does. In Figure 9.19, each data point represents the average working route length (in hops) of a test case network at the average nodal degree indicated, when working routing is via shortest paths (by span length). Data points for all 124 test case networks are shown in the same plot, but polynomial regression curves (quadratics in this case) are estimated for each network family individually. The lowest curve corresponds to the 15-node family, the next curve corresponds to the 20-node family, and so on. It is apparent from the plot that as expected, there is a very clear relationship between average shortest path working route length and $\bar{d}$, as all five curves have $R^2 \geq 0.9633$. In general, we can observe a trend that as $\bar{d}$ decreases, the increase in average working route lengths accelerates, particularly in larger networks (i.e., those with more nodes). A similar analysis of the percentage of working routes (from shortest path routing) that are three hops or shorter shows that it too is strongly related to $\bar{d}$, as shown in Figure 9.20. Like in Figure 9.19, data for all 124 test case networks are included in the same plot, with regression curves (in this case purely linear) calculated for each network family individually. The top curve corresponds to the smallest (15-node) network, the next curve corresponds to the 20-node network, and so on. The regression lines all have $R^2 \geq 0.9093$. As expected, the trend is that as $\bar{d}$ increases, so too does the percentage of working routes with three hops or fewer.

---

[18] The working routing used herein is based on actual span lengths (say in terms of kilometres) or by extension, the relative cost of each unit of capacity needed on each span. However, as long as span lengths don't vary too widely, working routing via shortest paths in hops will approximate working routing via shortest path lengths.

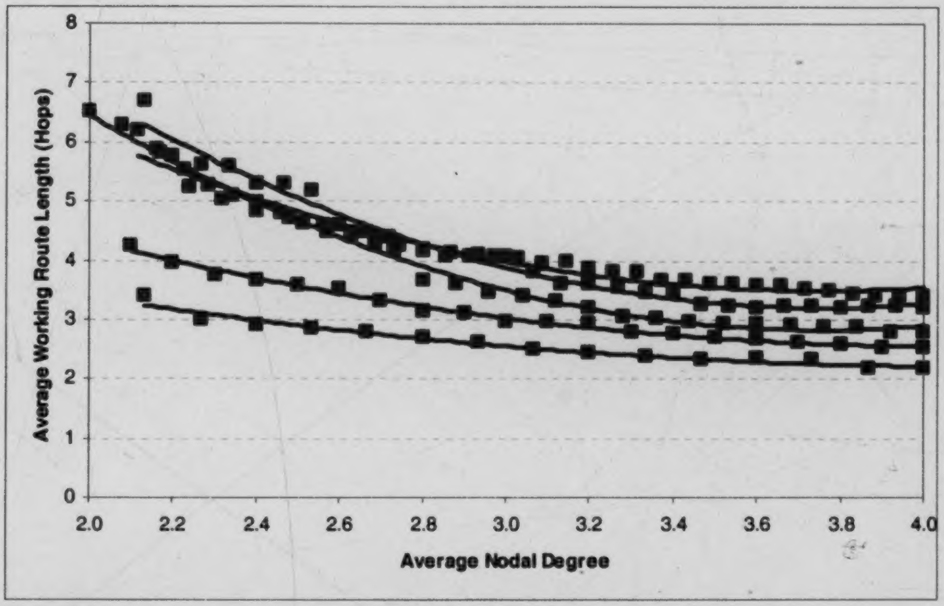


Figure 9.19 – Scatter plot of the average working route length (in hops) of shortest path working routing versus average nodal degree for all 124 test case networks.

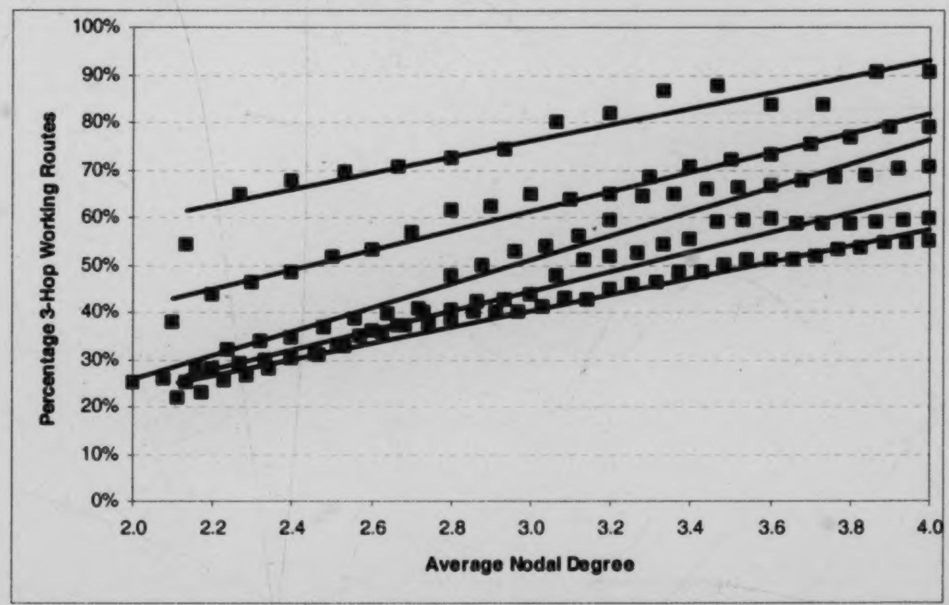


Figure 9.20 – Scatter plot of the percentage of working routes of three hops or less in shortest path working routing versus average nodal degree for all 124 test case networks.

While the above evidence alone doesn't confirm our hypothesis that the NISR-S gap ratio decreases with increases in $\bar{d}$ because of its dependence on average working route length and the number of working routes of three hops or shorter, seeing such strong relationships between those factors and $\bar{d}$ does add credibility to it. As an interesting aside, a scatter plot of the percentage of working routes that are three hops or shorter versus the average working route length in Figure 9.21 also shows a surprisingly compelling relationship between the two. The quadratic polynomial regression curve shown has $R^2 = 0.9906$, and it is quite apparent from the closely clustered data points that as the average working route length decreases, the percentage of working routes of three hops or less increases, and furthermore, that this increase accelerates (almost exponentially) with further decreases in average working route length.

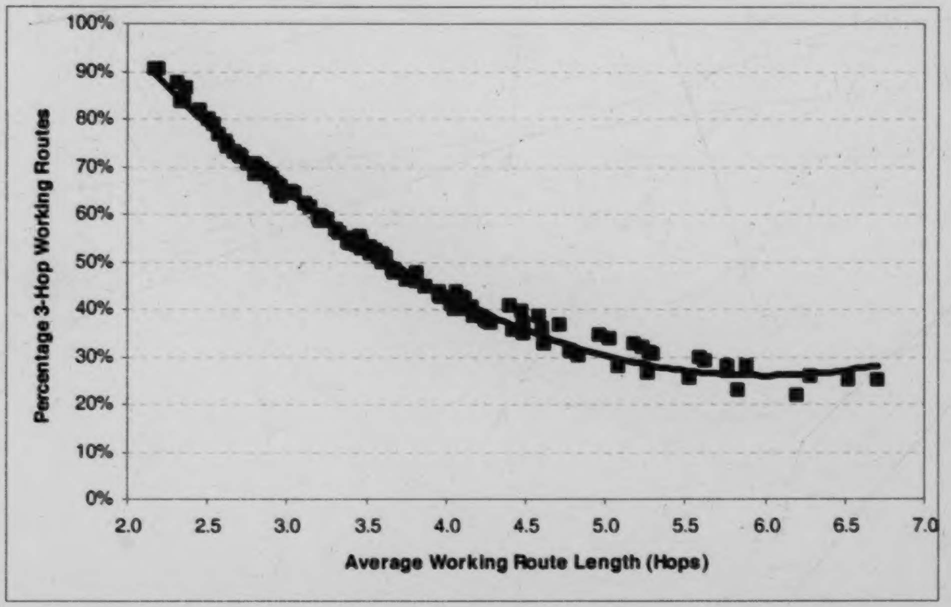


Figure 9.21 – Scatter plot of the percentage of working routes of three hops or less in shortest path working routing versus their average length (in hops) for all 124 test case networks.

The next logical course is now to investigate the direct relationship between the NISR-S gap ratio and the average working route length as well as the percentage of working routes that are three hops or shorter. Scatter plots of that data are shown in

Figure 9.22 and Figure 9.23, respectively, and estimated regression lines are drawn for each network family individually. In Figure 9.22, which plots the gap ratio against the average working route length, the regression line to the far left corresponds to the 15-node network family, the next line moving to the right corresponds to the 20-node family, and so on. While the 35n70s1 trend line has numerous outliers leading to a coefficient of determination of $R^2 = 0.7248$, all of the others compare to the actual data points quite strongly, with $0.8382 \leq R^2 \leq 0.9198$, and an average R^2 value of $\bar{R}^2 = 0.8570$. They also had an average correlation coefficient of $\bar{r}_{xy} = 0.9249$. By comparison, the gap ratio and $\bar{d}$ seemed to correlate a little less strongly, with $\bar{R}^2 = 0.8289$ and $\bar{r}_{xy} = 0.9098$ on average, suggesting that the average working route length seems to be a slightly more important factor in predicting the NISR-S gap ratio than average nodal degree itself. In Figure 9.23, which plots the NISR-S gap ratio against the percentage of working routes that are three hops or shorter, the rightmost regression line corresponds to the 15-node network family, and moving to the left, the network sizes (in nodes) become larger. The regression trend lines have $\bar{R}^2 = 0.8569$ and $\bar{r}_{xy} = 0.9253$, on average, further suggesting that the NISR-S gap ratio is also correlated to the percentage of working routes that are three hops or shorter slightly more so than it is to $\bar{d}$. And as we would expect from the observations of Figure 9.21, the gap ratio appears to be no more or less correlated to the average working route length as to the percentage of three-hop working routes since those two factors are themselves so closely associated. These analyses also provide additional evidence to suggest that our hypothesis that the length of the working routes is primarily responsible for NISR-S spare capacity requirements approaching path restoration in networks with high $\bar{d}$.

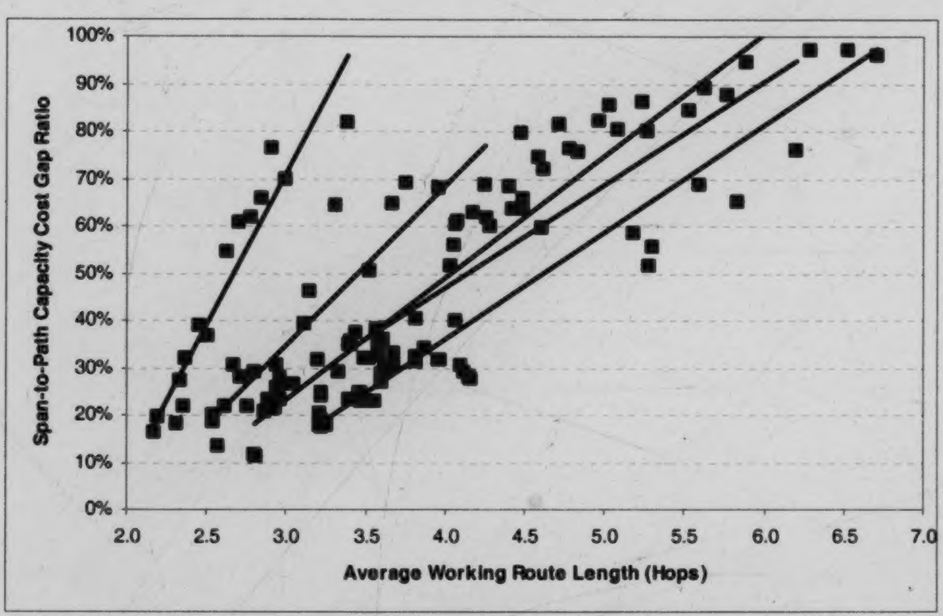


Figure 9.22 – Scatter plot of the gap ratios of NISR-S SCA capacity costs versus average working route lengths (in hops) for all 124 test case networks.

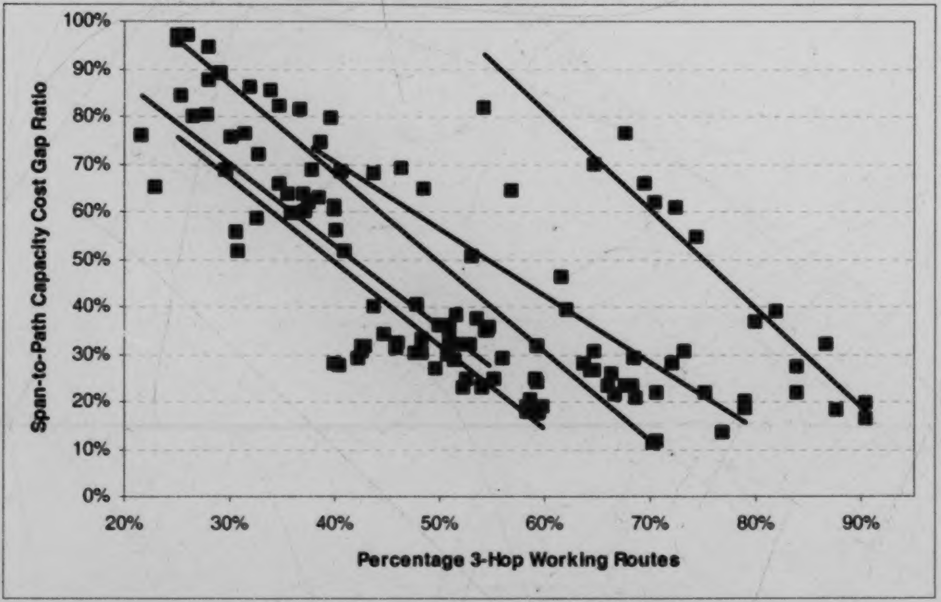


Figure 9.23 – Scatter plot of the gap ratios of NISR-S SCA capacity costs versus the percentage of working routes of three hops or shorter for all 124 test case networks.

9.4.1.2 NISR-S SPARE CAPACITY COST TREND LINE ANALYSIS

One final analysis we make regarding NISR-S spare capacity costs is an investigation of spare capacity cost trends. In Figure 9.8 through Figure 9.12, we plotted normalized spare capacity costs of NISR-S and the conventional benchmark survivability mechanisms versus average nodal degree, $\bar{d}$, for the five test case network families studied. It was quite clear that for all survivability mechanisms, spare capacity requirements decrease in a somewhat linear fashion with increases in $\bar{d}$, but no formal regression analysis was conducted. The five lines shown in Figure 9.24 are the estimated regression lines of the normalized spare capacity cost versus average nodal degree for the 25n50s1 family of test case networks. In other words, Figure 9.24 was generated by adding regression trend lines to Figure 9.10 and then removing the original data so that the trend lines stand out better. The top two trend lines correspond to the span-restorable and *p*-cycle network designs, respectively, while the bottom two represent SBPP and path-restorable designs, and the middle line corresponds to NISR-S designs. For all survivability mechanism, including NISR-S, the trend lines have very high coefficients of determination ($R^2 \geq 0.9093$ and $\bar{R}^2 = 0.9221$), suggesting very strongly linear relationships.

The most striking observation we make is that the trend lines for the four conventional benchmark survivability schemes (span restoration, *p*-cycles, SBPP, and path restoration) appear to have almost identical slopes, while the NISR-S trend line deviates significantly. We present only data for the 25n50s1 network family because that observation stands out most clearly for that family, but all network families show similar behaviour with the four benchmark survivability mechanisms exhibiting almost parallel trend lines and a NISR-S trend line that is substantially steeper in slope.

The interpretation is that there is something fundamentally unique about node-inclusive span restoration compared to the benchmark mechanisms, and that it produces a very different relationship between spare capacity requirements and $\bar{d}$. It is also interesting to note that the NISR-S spare capacity costs appear to shift from being very like those of span restoration to being very much like those of path restoration. This is all consistent with our other observations and discussion that in networks with high average nodal degree, and therefore relatively short working routes (with many at three hops or less), restoration routes in NISR-S will be very similar to those of path restoration. In fact the NISR-S trend line even crosses the path restoration trend line at high $\bar{d}$. Conversely, in networks with low $\bar{d}$, there will be a relatively smaller difference between restoration routes used by NISR-S and those used by span restoration.

Figure 9.25 and Figure 9.26 present the equivalent trend lines for spare capacity versus average working route length and the percentage of three-hop working routes, respectively, in the 25-node network family. Once again, the trend lines of the four conventional benchmark mechanisms all have nearly parallel slopes, while the NISR-S trend line differs considerably, which further supports the above conclusions.

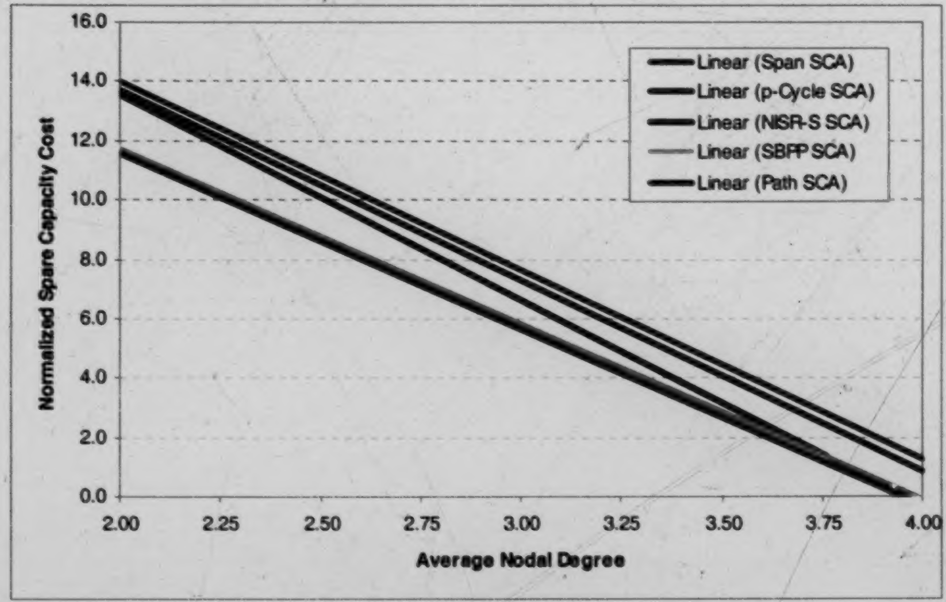


Figure 9.24 – Normalized SCA spare capacity cost linear regression trend lines plotted against average nodal degree for the 25n50s1 network family.

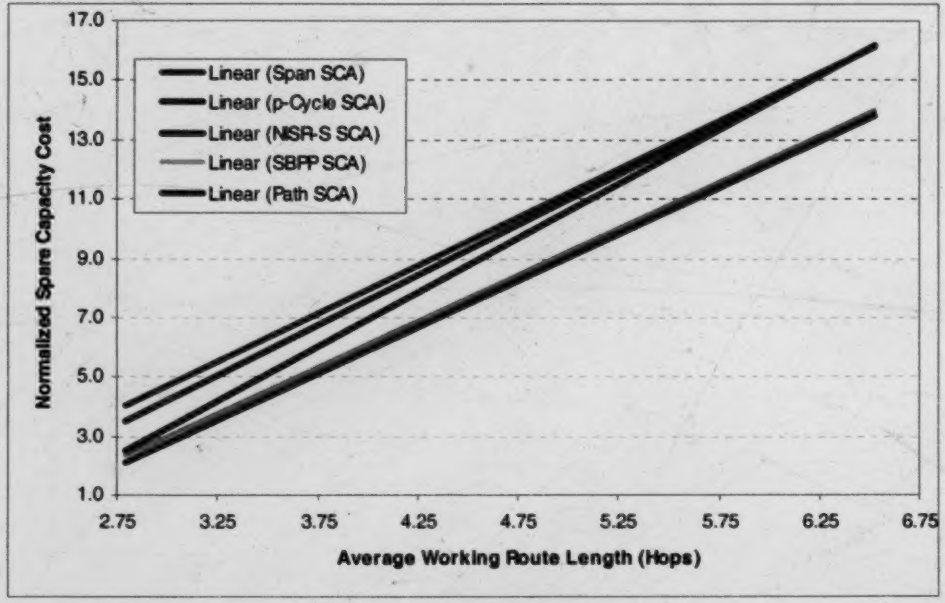


Figure 9.25 – Normalized SCA spare capacity cost linear regression trend lines plotted against average working route length for the 25n50s1 network family.

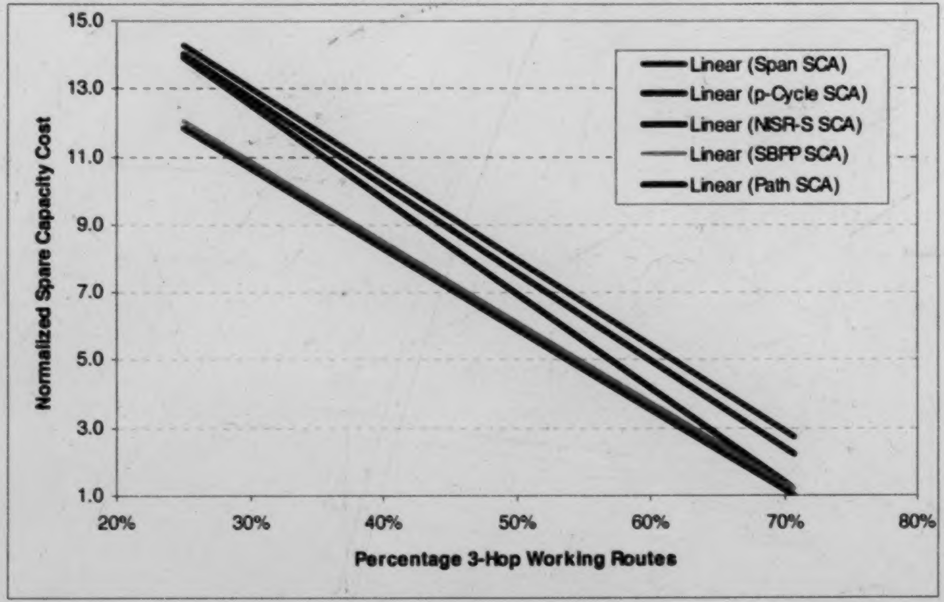


Figure 9.26 – Normalized SCA spare capacity cost linear regression trend lines plotted against the percentage of three-hop working routes for the 25n50s1 network family.

9.4.2 NISR-S REDUNDANCY

Figure 9.27 through Figure 9.31 show capacity cost redundancies of optimally designed networks of the various families designed to be 100% restorable using NISR-S, as well as the equivalent designs using the SCA models for span restoration, *p*-cycle restoration, SBPP, and path restoration. Like the capacity cost figures already seen, each figure provides data for a single network family, and each data point represents the R_{cost} of the optimal solution for the member of the family with the indicated average nodal degree ($\bar{d}$) using the specified survivability mechanism. The $1/(\bar{d}-1)$ lower bound on capacity redundancy in a span-restorable network has been added to each figure for reference and comparison purposes.

Capacity cost redundancy curves of the NISR-S SCA designs mirrored those for spare capacity, with NISR-S requiring an average of 19.7% less spare capacity than span restoration, 17.1% less than *p*-cycle restoration, 13.8% more than SBPP, and 20.5% more than path restoration. The most noteworthy aspect of these curves is how closely they approach the $1/(\bar{d}-1)$ lower bound on capacity redundancy in a span-restorable network. It doesn't quite reach it, but the bound will not necessarily hold for NISR-S networks anyway since it was developed for a span-restorable network. The NISR-S redundancy curves could therefore break below it, and in preliminary studies of the JCA variant of NISR, they actually do for approximately half of our test case networks. Recall in the derivation of the $1/(\bar{d}-1)$ lower bound in Section 4.7 that the basis for the bound is that there must be sufficient spare capacity on the spans adjacent to the failed span because restoration routing must be performed between the end-nodes of the failed span itself. However, in NISR-S, restoration routes are formed between custodial nodes one hop removed from the failed span, effectively spreading the spare capacity requirement out to a greater number of surrounding spans, and even allowing the use of stub-released capacity as well. Moreover, from the point of view of failures on short working routes, NISR-S closely approximates path restoration, and in the case of 3-hop working routes, NISR-S will actually be identical to path restoration for a failure of the middle span.

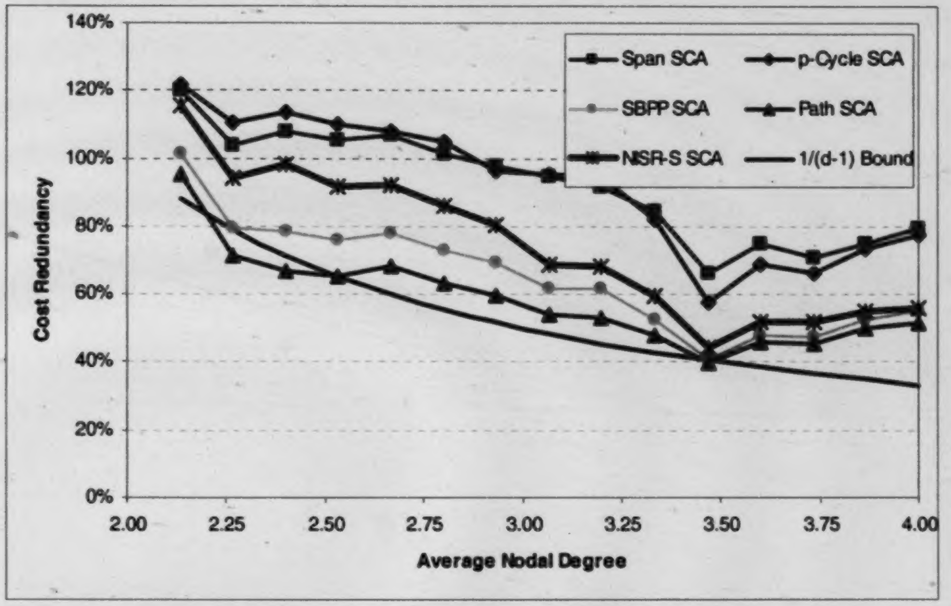


Figure 9.27 – NISR-S SCA capacity cost redundancy for the 15n30s1 network family.

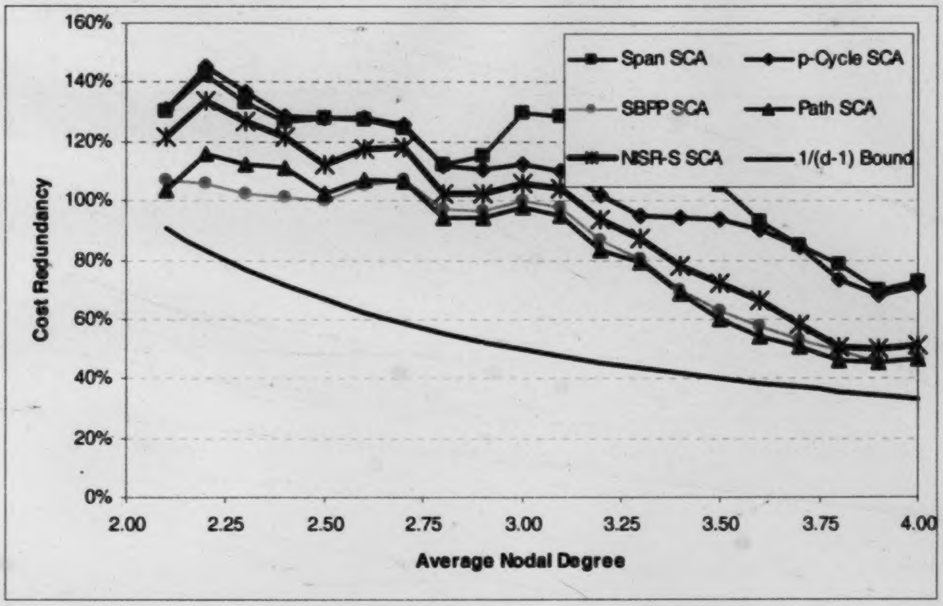


Figure 9.28 – NISR-S SCA capacity cost redundancy for the 20n40s1 network family.

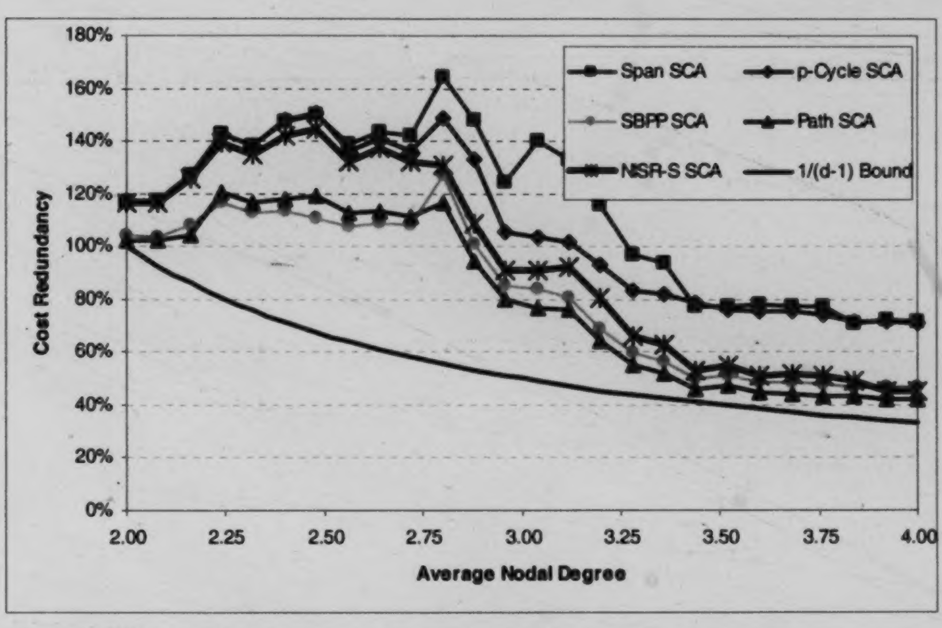


Figure 9.29 – NISR-S SCA capacity cost redundancy for the 25n50s1 network family.

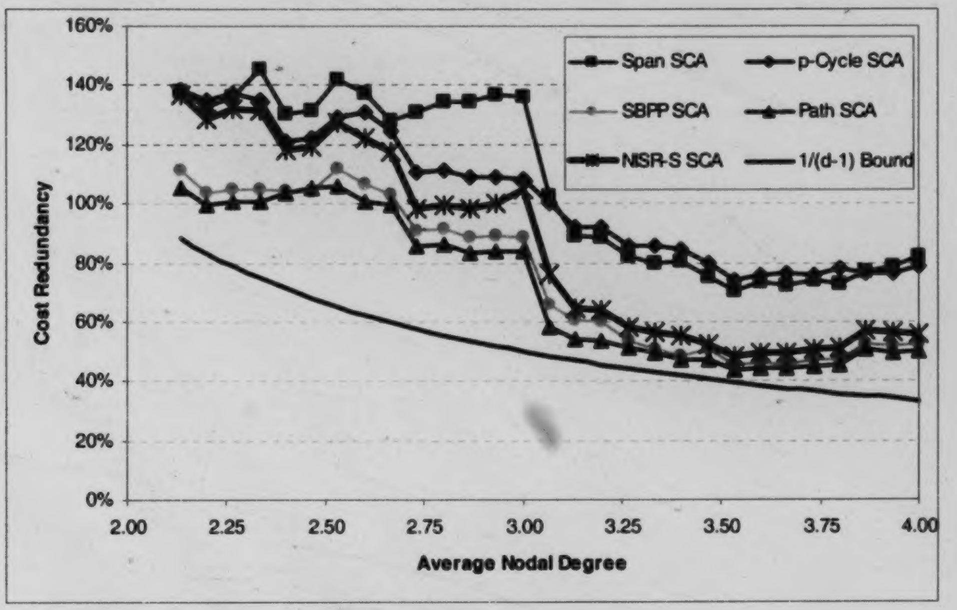


Figure 9.30 – NISR-S SCA capacity cost redundancy for the 30n60s1 network family.

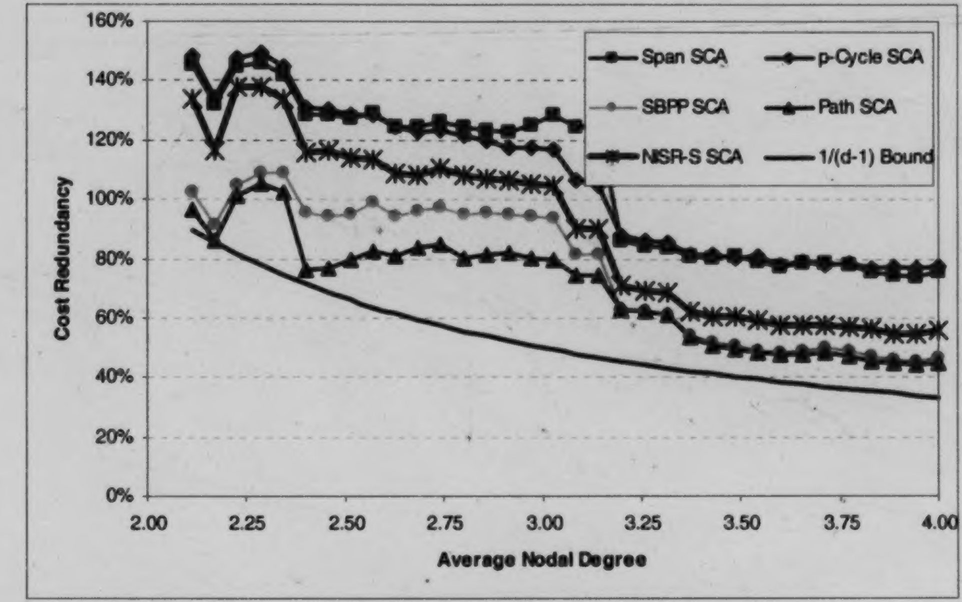


Figure 9.31 – NISR-S SCA capacity cost redundancy for the 35n70s1 network family.

9.4.3 USING NISR FOR NODE-FAILURE RESTORATION

As we discussed earlier in Section 9.1, node-inclusive span restoration can be used to restore node failures in addition to span failures. In Sections 9.2.2 and 9.2.3 we developed specific ILP models for optimally designing networks to do just that, and now compare the costs of providing node-failure restoration to the cost of providing only span-failure restoration. In Figure 9.32 through Figure 9.36, we plot the normalized spare capacity costs of networks designed to be 100% restorable with the SCA ILP optimization models for NISR-S, NISR-N, and NISR-NS. Each figure provides data for a single network family, and each data point represents the total wavelength-kilometres of spare capacity required to provide for full restoration in the optimally-designed solution for the member of the family with the indicated average nodal degree ($\bar{d}$) using the specified design model. Since all models are SCA models, working capacity is determined via shortest path routing. In each figure, capacity costs have been normalized to the same lowest-cost design over all networks in the family and all survivability mechanisms as we used in Section 6.4. Within each figure, data points have been organized into three curves, one for each design model.

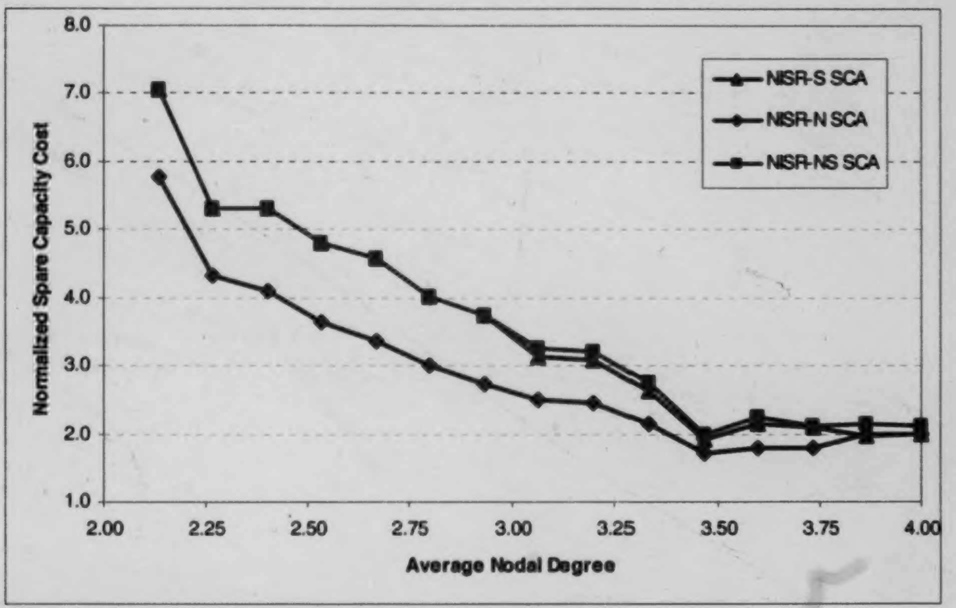


Figure 9.32 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 15n30s1 network family.

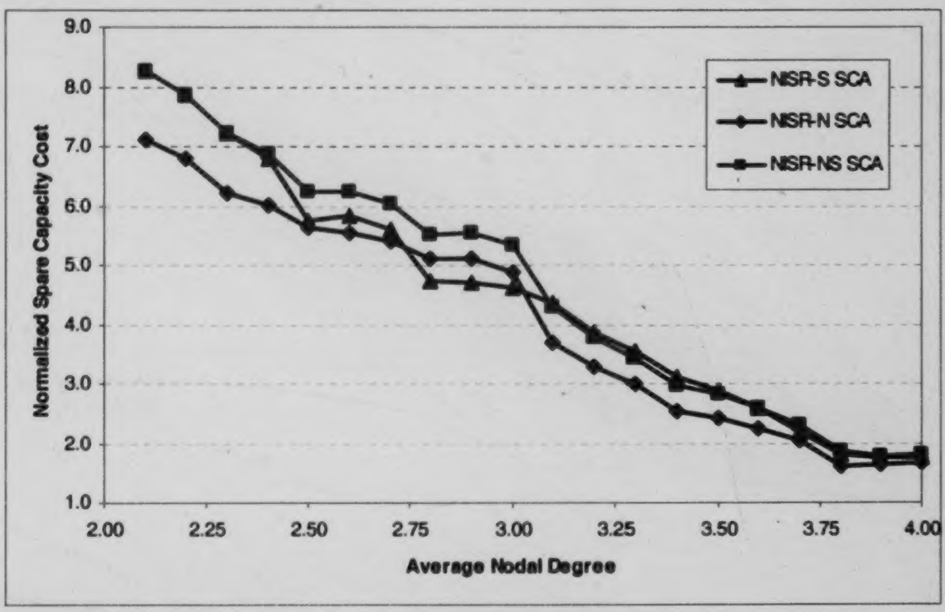


Figure 9.33 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 20n40s1 network family.

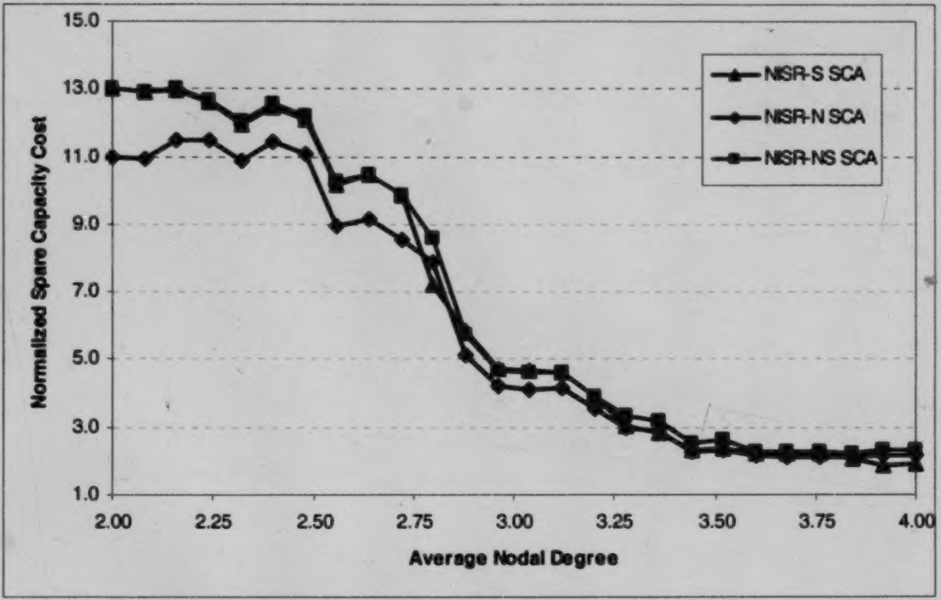


Figure 9.34 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 25n50s1 network family.

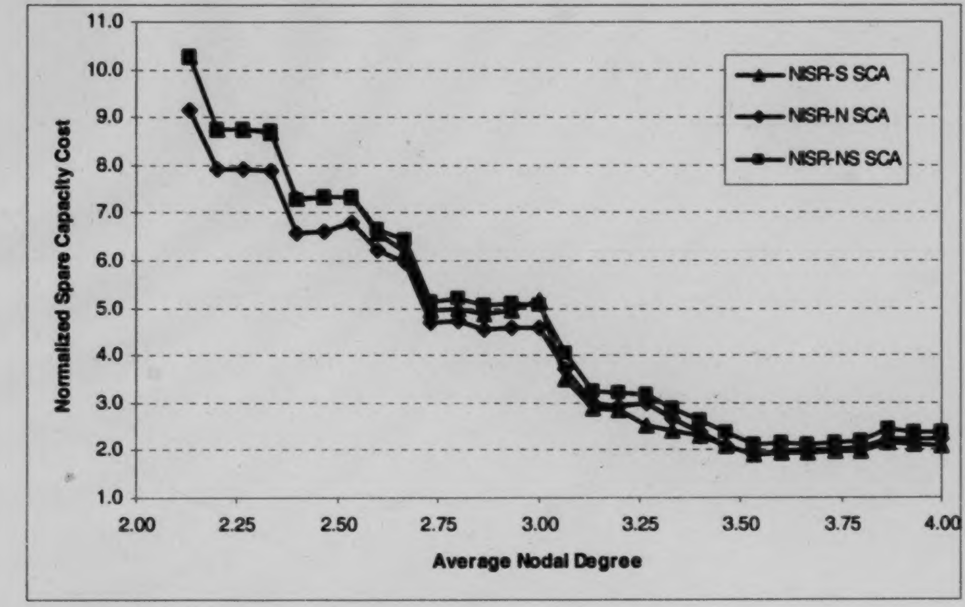


Figure 9.35 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 30n60s1 network family.

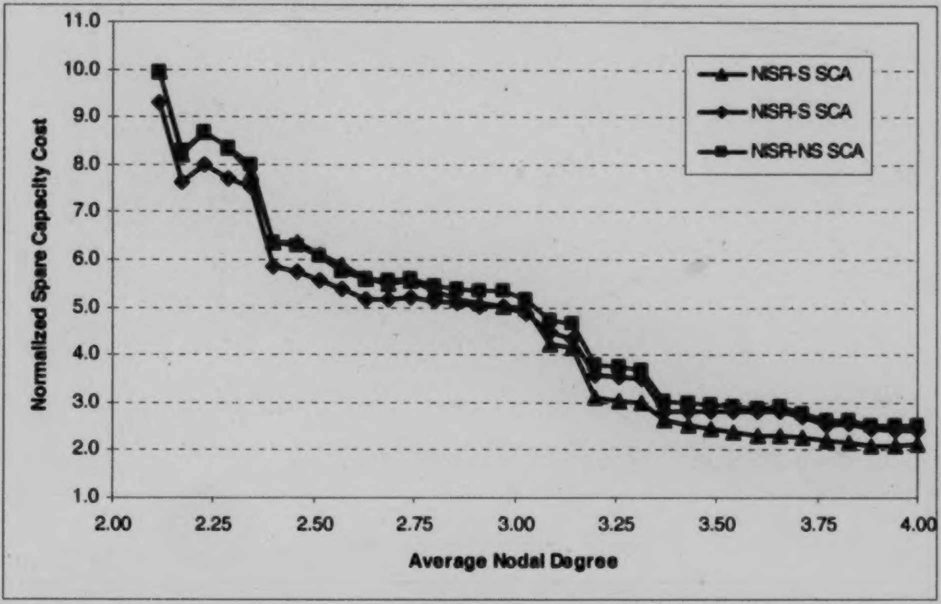


Figure 9.36 – NISR-S SCA, NISR-N SCA, and NISR-NS SCA normalized spare capacity costs for the 35n70s1 network family.

Comparisons of the three variants of NISR show that for most test case networks, it costs very little additional spare capacity to provide both span-failure and node-failure restoration (NISR-NS – the red curve) than it does to provide only span-failure restoration (NISR-S – the green curve). On average, NISR-NS needed 6.7% more spare capacity than NISR-S, but in networks with $\bar{d} < 3.0$, the difference was 2.0%, and in networks with $\bar{d} < 2.8$, it was only 0.9%. In other words, it costs more spare capacity to provide both span-failure and node-failure restoration (relative to the span-failure only design) in highly connected networks than it does in sparse networks. One possible reason is that in sparse networks, there is a greater number of lengthy chains (see CHAPTER 7). Even with NISR, there are some loop-back spare capacity requirements within a chain, and whether it is a node or a span that fails within the chain, any working flow through it will still need to be looped back to the anchor nodes. Conversely, in richly connected networks, working routes are relatively short and cross very few (and very short) chains, so the above loop-back effect is not nearly as dominant. Additionally, in such networks, a node failure truly manifests itself as multiple span failures since each node on average will be connected to a greater number of spans.

We also notice that the additional costs for providing node-failure restoration (i.e., the differences between NISR-NS and NISR-S) are greater in larger networks (i.e., those with more nodes). For instance, in the 15-node network family, the average difference between NISR-NS and NISR-S spare capacity costs is only 2.5%, in 20n40s1 it is 3.7%, for 25n50s1 it is 4.5%, in 30n60s1 it is 7.8%, and in the 35-node family, the difference is 10.9%. This is due mainly to combined effects of two factors. The first is that since every node pair exchanges demands, the number of demands in our demand models is $O(N^2)$. The second issue is that with failure of a node, lightpaths originating or terminating at that node are inherently unrestorable and so the design model simply performs stub-release of their working capacity along their *entire* working routes. So even though failure of a node in any network will result in $N-1$ sets of lightpaths disappearing from the restoration problem (each node has that many demands terminating or originating at it), the total number of demands will be $N \cdot (N-1)/2$, meaning on a proportional basis, a $2/N$ fraction of the demands in a network will be dropped entirely from the network. In a 15-node network then, $2/15 = 13.3\%$ of all demands will be dropped in the event of a node failure, and so on average, 13.3% of the working capacity in the network will be stub released and could be used as spare capacity to restore any other lightpaths affected by the node failure. On the other hand, in a 20-node network, only 10% of the working capacity will be stub released in that fashion. In a 25-node node network, the percentage drops to 8%, in a 30-node network it is 6.7%, and in a 35-node network it is 5.7%.

Another interesting finding is that if we provide *only* node-failure restoration with the NISR-N design model (no span-failure restoration is explicitly required), then over all 124 test case networks, it costs an average of 5.2% *less* spare capacity than span-failure restoration only (NISR-S). However, this too is greatly dependent on the connectivity and number of nodes in a network. In networks with $\bar{d} < 3.0$, the difference was 10.9% and in networks with $\bar{d} < 2.8$, it was 12.1%. In 15-node networks, the difference was 22.5%, in 20-node networks it was 9.7%, in the 25n50s1 family it was 6.5%, in the 30n60s1 family the difference was only 2.0%, and in the 35-node networks it was actually 3.3% *more* costly (in terms of spare capacity) for NISR-N that it was for NISR-S. The reasons are the same as those discussed above in the comparison between NISR-S and NISR-NS. In small networks, a higher proportion of working lightpaths are released in the event of a node failure, and so a greater amount of working capacity is available for reuse as spare capacity. As a consequence, the cost of providing only node-failure restoration is significantly less than the cost of providing span-failure restoration. In larger networks, only a very small proportion of the total working capacity is affected by stub-release, and so a greater amount of additional spare capacity is required relative to NISR-S. This effect is so dominant that in the largest network family, it costs more on average to provide node-failure restoration than it did for span-failure restoration. This is completely contrary to what we observed in all of the smaller network families.

The significance of these findings is that they suggest that a network planner might choose not to provide node-failure restoration in particularly richly connected or large networks. However, in a North American network that might typically have $\bar{d} < 2.8$, or in a metropolitan network, with not many nodes, it is relatively inexpensive to provide node-failure restoration in addition to span-failure restoration.

Alternatively, a planner could design a network to be fully span-failure restorable using NISR-S, but allow best-efforts node-failure restoration that will be inherent in the NISR mechanism. Even though spare capacity is optimized and installed for span-failure restoration only, at least partial node-failure restoration will still be possible for any given node failure scenario. A simple modification of the NISR-N ILP model in Section 9.2.2 can be used to test the node-failure restorability levels. First, we need to introduce more new notation as follows:

New Decision Variables:

- $u_n^r \geq 0$ is the amount of working flow for demand relation *r* that is not restorable when node *n* fails.

The first change we make to the NISR-N model is that the spare capacity variables (s_j) become input parameters whose values are taken from the solution to the NISR-S design model. These s_j values represent the optimal spare capacity installed to make the network fully span-failure restorable using the NISR survivability mechanism. The constraint set in equation (9.10), which ensures that there is sufficient restoration flow to *fully* restore all of the lightpaths affected by the failure of node n, is converted to equation (9.17). The addition of the un-restorability variable, u_n^r, allows some of the working flow in $g^{r,q}$ to not be restored if there isn't sufficient spare capacity to do so. The objective function is then modified to equation (9.18), which minimizes the average percentage of lightpaths that cannot be restored in the event of a node failure.

$$\sum_{\forall p \in \tilde{P}_{n,m}^{r,q}} \tilde{f}_{n,m}^{r,q,p} + u_n^r = g^{r,q} \qquad \begin{array}{c} \forall n \in N \quad \forall r \in \tilde{D}_n \\ \forall q \in \tilde{Q}_n^r \middle| O_r \neq n \neq T_r \\ \forall m \in \mathrm{N} \middle| m = \tilde{N}_n^{r,q} \end{array} \qquad (9.17)$$

$$\text{Minimize} \quad \frac{\sum_{\forall n \in N} \sum_{\forall r \in \tilde{D}_n} \frac{u_n^r}{d^r}}{|N| \cdot |\tilde{D}_n|} \qquad (9.18)$$

Solutions of this ILP model show that in all test case networks, only a very small proportion of working lightpaths will not be restorable in the event of any node failure. Figure 9.37 is a scatter plot of node-failure un-restorability levels in networks designed to be fully span-failure restorable using NISR-S. Each data point corresponds to one of the test case networks with the indicated average nodal degree. Linear regression trend lines are drawn individually for each test case network family as well as the full suite of 124 test case networks as a whole. The bottom black trend line corresponds to the 15n30s1 network family, the next corresponds to the 20n40s1 network family, and so on, and the blue trend line (third from the top) corresponds to the entire set of 124 test case networks. The general overall trend that emerges from the data is that a greater proportion of potentially survivable working lightpaths will not be restorable in larger networks (in terms of the number of nodes) and richly connected networks than in smaller sparse networks. By "potentially survivable", we mean to say that we do not consider lightpaths that originate or terminate at the failed node since those lightpaths are strictly failed and unresrtorable in any case, no matter the restorability mechanism or amount of spare capacity. On average over all 124 test cases, only 1.51% of all potentially survivable working lightpaths are not restorable. In the 15n30s1 network family, the average un-restorability is 0.64%, in the 20-node networks it is 1.01%, in the 25n50s1 networks it is 1.34%, in 30n60s1 the average un-restorability is 1.84%, and in the 35-node family it is 2.04%. And even in the largest network families, sparse networks are fully or nearly fully node-failure restorable. In fact, 20 of the 124 test case networks had stricly no node-failure un-restorability whatsoever, 35 had less than 0.5% un-restorability, and 53 of them had less than 1% un-restorability. In other words, in $53/124 = 42.7\%$ of the test case networks designed with NISR-S, 99% or more of the potentially survivable working lightpaths can be expected to be restorable to some random node failure.

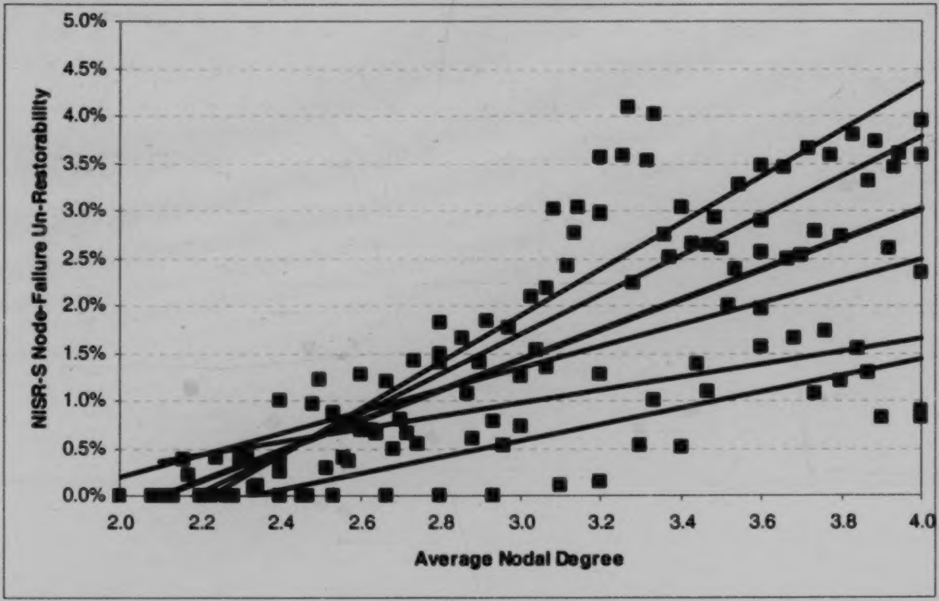


Figure 9.37 – Node-failure un-restorability levels (excluding terminating lightpaths) of all 124 test case networks as designed with the NISR-S optimization model.

9.5 CLOSING REMARKS

Span restoration is a highly localized and easily implemented network survivability mechanism, but by its very nature, it cannot be used to restore node failures. Path restoration, on the other hand, is significantly more capacity efficient and can be used to restore node failures. However, potentially very many node pairs may be involved simultaneously and the real time signalling loads and capacity contention imposed on the network's nodes lead to much greater average delay and longer average restoration route lengths than in span restoration. Node-inclusive span restoration is a new survivability mechanism based on conventional span restoration, and combines positive aspects of both span restoration and path restoration.

NISR provides both span-failure and node-failure restoration, while also achieving capacity efficiencies approaching those of path restoration. On average, NISR networks need 19.7% less spare capacity than span-restorable networks, and in networks with high $\bar{d}$, NISR spare capacity requirements are significantly closer to those of path restoration than they are to those of span restoration. We showed that in sparse networks, spare capacity requirements in NISR networks were similar to (but slightly smaller than) those of span-restorable networks, and that as the average nodal degree of a network increases, NISR spare capacity requirements become more and more like those of path restoration. Analyses show that in richly connected networks, shortest path working routing results in quite short working lightpaths, and for those lightpaths that are three-hops or shorter, NISR is actually indistinguishable from path restoration. In contrast, working routes are much longer in sparse networks, and so NISR acts more like span restoration than it does like path restoration. Regression analysis also showed that the benchmark survivability mechanisms all tend to behave similarly to one another with respect to changes in spare capacity requirements as $\bar{d}$ increases, but NISR does not. Again, the reason appears to be that NISR efficiency is affected by working route lengths, while the other mechanisms are not (at least not to the same degree).

When networks are explicitly designed to be fully NISR restorable to node failures in addition to span failures, spare capacity costs are still approximately 15% below span-restorable network designs, and are only 6.7% above the NISR-S designs (where only span-failure restoration is assured). And because of an interesting interaction of factors, large and/or dense networks require significantly more additional spare capacity to assure full node-failure restoration than small and/or sparse networks. Since failure of a node means any working lightpaths originating or terminating at the node are strictly not restorable, working capacity used by such lightpaths is released and made available for spare capacity. In sparse networks, those lightpaths tend to be longer than in dense networks, and in small networks, they tend to make up a greater proportion of all lightpaths than in large networks. So in networks with fewer nodes or a smaller average nodal degree, a relatively substantial amount of its working capacity is useable as spare capacity when a node fails. Consequently, less additional spare capacity needs to be added to those networks to deal with node failures. In cases when a network operator may not wish to (or is unable to) add spare capacity to guarantee full node-failure restoration, NISR will still be able to provide node-failure restorability for most working lightpaths anyway. In fact, nearly half of all test case networks we studied had better than 99% average node-failure restorability over all working lightpaths.

9.5.1 CONSIDERATIONS FOR DYNAMIC SERVICE PROVISIONING

Although span restoration may be simpler to implement, its capacity requirements are greater than those of NISR, and it is unable to provide node-failure recovery. So NISR shows clear advantages over span restoration. Path restoration and SBPP, on the other hand, are both able to provide node-failure recovery, and their capacity efficiencies are actually better than NISR. However, they also possess disadvantages that NISR does not. The main advantage over path restoration is that, although NISR requires solution of a MCMF-type restoration process like path restoration, it does so with a much smaller number of node pairs, and is highly localized by comparison. But what does NISR provide to a network operator that SBPP does not?

First, based on observations from [16] and [23], it is almost certain that NISR dual-failure availability will be higher than SBPP, especially for premium services. We say this because the NISR mechanism is more localized and because protection of a working path in NISR is inherently segmented and failure-specific (the lengthy end-to-end backup paths used by SBPP will, by comparison, result in a greater number of outage-causing dual failure combinations). Segmentation is a strong principle in enhancing availability as shown by work in [101]. But the main advantage is seen if we consider the implications of highly dynamic demand provisioning. The path provisioning process under SBPP must explicitly arrange the shared backup path at the time of each service set-up, which requires a global OSPF-TE type of topology and resource database, including all backup route and capacity-sharing data. However, under the PWCE provisioning paradigm used by span restoration and its derivatives (see Section 4.5), protection is inherent so long as the capacity is present to route the new path. We are not required to arrange a backup path for every individual lightpath provisioning operation or even to update a network-wide state for each new protection path set-up or takedown. Any capacity used for provisioning is itself protected so any service provisioned over available working capacity is also protected with no further action. Since it is individual spans that are protected, we require only that each span crossed by a newly provisioned lightpath possesses at least one remaining protected channel. Provisioning a new path is solely a matter of routing the working path and designating its protection status, with no concern for protection arrangements because the envelope of working capacity on each span is itself protected as an automatic function of the embedded restoration mechanism.

9.5.2 FUTURE DIRECTIONS

At present we limit ourselves to considering the fundamental capacity design characteristics of a network using the NISR concept. An interesting area for further work is the development and testing of a specialized version of a self-organizing distributed restoration protocol of the type in [47] or [62] intended for NISR use. A dynamic provisioning protocol could also be developed in tandem, and would ideally allow more efficient use of installed spare capacity and possibly even maximize node-failure restorability. Implementation of the protocols within OPNET Modeler [90] or within the TRnet [112] experimental test-bed network would allow their characterization and provide valuable insights into their operation.

Preliminary testing of a JCA variant of the NISR network design models has indicated that NISR capacity redundancies can easily reach below the $1/(\bar{d}-1)$ lower bound on capacity redundancy in a span-restorable network, behaviour that was suspected but not observed in NISR SCA designs. A complete analysis of the effects of jointly optimizing working routing in an NISR network may also expose additional insights and/or possibly provide further supporting evidence for concepts or behaviour we've already discussed. For instance, if we found that working routes were significantly shorter in NISR JCA designs than they are in equivalent span-restorable JCA designs, it would support our theory that working route lengths are a key factor in determining how much more efficient NISR is than span restoration.

One final area we can investigate is NISR dual-failure restorability. In the previous sub-section, we discussed the likelihood that NISR dual-failure restorability levels are better than SBPP. While the findings of the referenced works strongly suggests that this is so, a thorough analysis could easily provide confirmation.

CHAPTER 10
CLOSING DISCUSSION

10.1 SUMMARY OF THESIS

The main intent of this thesis is to provide network planners with a better understanding of the fundamentals of mesh-network survivability, and to document several new methods for protecting and/or designing transport networks.

We opened the thesis with a review of relevant set theory, graph theory, and operations research in CHAPTER 2 and a review of transport networks and key enabling technologies in CHAPTER 3. We provided an explanation of network survivability methods in CHAPTER 4, and developed the mathematical formulations for the corresponding network design models in CHAPTER 5.

In CHAPTER 6, we provided thorough analyses and comparisons of various benchmark restoration and protection mechanisms in terms of overall capacity requirements and restoration capacity efficiency. The schemes we studied were 1+1 automatic protection switching, span restoration, *p*-cycle restoration, shared backup path protection, and path restoration, including SCA and JCA design variants. Useful insights and understanding were gained in how the various mesh schemes behave over varying network connectivity. Two key findings of note were (1) there is an identifiable and significant drop in capacity requirements in networks in the vicinity of $\bar{d} = 3.0$, and (2) due to the effects of stub-release, path-restorable JCA designs have capacity redundancies that can approach 0%.

In CHAPTER 7, we developed the meta-mesh concept for span-restoration in sparse mesh networks. We showed that increases in capacity efficiency could be gained by targeting chains of degree-2 nodes, and allowing the restoration mechanism to fail express traffic to the end-nodes of the chain, reducing the loop-back capacity requirements. By treating express wavelengths entirely with mesh-based restoration principles in the network outside the chain, their working capacities never enter into the spare capacity sizing of the chain, thereby reducing overall spare capacity requirements and increasing capacity efficiency. Meta-mesh capacity efficiency in quirements and increasing capacity efficiency. Meta-mesh capacity efficiency in sparse networks approaches that of path restoration, but survivability is via a much simpler version of span restoration. Interestingly, in spite of the fact that meta-mesh is a form of span restoration, redundancies of meta-mesh networks are able to drop below the $1/(\bar{d}-1)$ lower bound on redundancy in a span-restorable network.

In CHAPTER 8, we developed ILP models for the complete "green-fields" topological network design problem. A fast 3-step heuristic approach is shown to produce near-optimal designs much more quickly than the full problem, and for large test case networks, yields good designs where the full problem is unable to even provide a feasible solution in a reasonable time. In many test cases, the heuristic was able to significantly outperform the full problem in terms of both runtime and solution quality.

CHAPTER 9 introduced node-inclusive span restoration, which is a modification of span restoration that is able to provide full node-failure restoration while simultaneously reducing spare capacity requirements. The key idea is that we perform restoration between custodial nodes that are one hop removed from the end-nodes of a failed span along the path of the original working route of the affected demand. Essentially, this is equivalent to a compromise between span restoration and path restoration in the sense that in NISR, the custodial nodes are further out than they would be in span restoration but not so far out that they reach the end-nodes of the demand like they would in path restoration.

10.1.1 MAIN CONTRIBUTIONS

There are four main contributions of this thesis:

1. Comparison of five common mesh survivability schemes
    - Performed analyses under varying graph connectivities
    - Compared capacity requirements and redundancy
    - Compared SCA versus JCA designs
    - Identified a significant drop in capacity costs at $\bar{d} \cong 3.0$
    - Path-restorable JCA designs can approach 0% capacity redundancy

2. Meta-mesh network design
    - Targeted degree-2 chains and eliminated loop-back spare capacity required for express traffic
    - Reductions in capacity requirements peaked in networks with $\bar{d} \cong 2.5$
    - Identified additional savings from reductions in spare wavelength channel counts (average of 19%) and working channel counts (28%)
3. Topological transport network design
    - Developed a green-fields network design model (MTRS)
    - Developed a heuristic approach to the MTRS problem
    - Reduced MTRS runtimes of many hours or even days to heuristic runtimes of minutes or hours
4. Node-inclusive span restoration
    - Developed three variants (span-failure restoration, node-failure restoration, and both span-failure and node-failure restoration)
    - Spare capacity requirements closely approached those of path restoration, especially for networks with high $\bar{d}$
    - Identified working route lengths as a key factor in NISR effectiveness
    - Showed that node-failure restoration is free for most demands

10.2 OTHER CONTRIBUTIONS OF PH.D. WORK

Besides the contributions presented and discussed herein, the overall Ph.D. work associated with this thesis made several other notable contributions. A total of 16 peer-reviewed publications, one book chapter, and four patent applications were produced, as well as numerous technical reports and presentations, as follows.

10.2.1 JOURNAL PAPERS

1. J. Doucette, W. D. Grover, "Shared-Risk Logical Span Groups in Span-Restorable Optical Networks: Analysis and Capacity Planning Model," *Photonic Network Communications*, vol. 9, no. 1, pp-35-53, January 2005.

2. J. Doucette, M. Clouqueur, W. D. Grover, "On the Availability and Capacity Requirements of Shared Backup Path-Protected Mesh Networks," *Optical Networks Magazine, Special Issue on Engineering the Next Generation Optical Internet*, vol. 4, no. 6, pp. 29-44, November/December 2003.

3. W. D. Grover, J. Doucette, M. Clouqueur, D. Leung, D. Stamatelakis, "New Options and Insights for Survivable Transport Networks," *IEEE Communications Magazine*, vol. 40, no. 1, pp. 34-41, January 2002.

4. W. D. Grover, J. Doucette, "Design of a Meta-Mesh of Chain Sub-Networks: Enhancing the Attractiveness of Mesh-Restorable WDM Networking on Low Connectivity Graphs," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 20, no. 1, pp. 47-61, January, 2002.

5. W. D. Grover, J. Doucette, "Topological design of span-restorable mesh transport networks," *Annals of Operations Research, Special Issue on Topological Design of Telecommunication Networks*, vol. 106, pp. 79-125, September 2001.

6. J. Doucette, W. Grover, "Influence of Modularity and Economy-of-scale Effects on Design of Mesh-Restorable DWDM Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol.18, no.10, pp. 1912-1923, October 2000.

10.2.2 PEER-REVIEWED CONFERENCE PAPERS

1. J. Doucette, D. He, W. D. Grover, O. Yang, "Algorithmic Approaches for Efficient Enumeration of Candidate *p*-Cycles and Capacitated *p*-Cycle Network Design," *Proceedings of the 4th International Workshop on Design of Reliable Communication Networks (DRCN 2003)*, Banff, AB, Canada, pp. 212-220, 19-22 October 2003.

2. J. Doucette, W. D. Grover, "Node-Inclusive Span Survivability in an Optical Mesh Transport Network," *Proceedings of the 19th Annual National Fiber Optic Engi-*

neers Conference (NFOEC 2003), Orlando, FL, USA, pp. 634-643, 7-11 September 2003.

3. J. Doucette, W. D. Grover, "Maintenance-Immune Optical Mesh Network Design," *Proceedings of the 18th Annual National Fiber Optic Engineers Conference (NFOEC 2002)*, Dallas, TX, USA, pp. 2049-2061, September 2002.
4. J. Doucette, W. D. Grover, "Capacity Design Studies of Span-Restorable Mesh Networks with Shared-Risk Link Group (SRLG) Effects," *Proceedings of the Optical Networking and Communications Conference (OptiComm 2002)*, Boston, MA, USA, pp. 25-38, July-August 2002.
5. W. D. Grover, J. Doucette, "Advances in Optical Network Design with *p*-Cycles: Joint Optimization and Pre-Selection of Candidate *p*-Cycles," *Proceedings of IEEE/LEOS Summer Topicals 2002*, Mont Tremblant, PQ, Canada, pp. 49-50 (paper WA2), July 2002.
6. W. D. Grover, J. Doucette, "A Novel Heuristic for Topology Planning and Evolution of Optical Mesh Networks," *Proceedings of IEEE Global Telecommunications Conference (GlobeCom 2001)*, San Antonio, TX, USA, pp. 2169-2173, November 2001.
7. J. Doucette, W. D. Grover, "Comparison of Mesh Protection and Restoration Schemes and the Dependency on Graph Connectivity," *Proceedings of the 3rd International Workshop on Design of Reliable Communication Networks (DRCN 2001)*, Budapest, Hungary, pp. 121-128, October 2001.
8. W. D. Grover, J. Doucette, "Increasing the Efficiency of Span-Restorable Mesh Networks on Low-Connectivity Graphs," *Proceedings of the 3rd International Workshop on Design of Reliable Communication Networks (DRCN 2001)*, Budapest, Hungary, pp. 99-106, October 2001.
9. J. Doucette, W. D. Grover, T. Bach, "Bi-criteria studies of mesh network restoration path-length versus capacity tradeoffs," *Proceedings of OSA Optical Fiber Communications Conference and Exhibit (OFC 2001)*, Anaheim, CA, USA, pp. TuG2-1 - TuG2-3, March 2001.

10. J. Doucette, W. D. Grover, R. Martens, "Modularity and Economy-of-Scale Effects in the Optimal Design of Mesh-Restorable Networks," *Proceedings of 1999 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE 1999)*, Edmonton, AB, Canada, pp. 226-231, May 1999.

10.2.3 BOOK CHAPTER

1. W. D. Grover, J. Doucette, D. Leung, A. Kodian, A. Sack, M. Clouqueur, G. Shen, W. Sukcharoenkana, "Design of Survivable Networks Based on *p*-Cycles", to appear in the *Handbook of Optimization in Telecommunications*, P.M. Pardalos, M. G. C. Resende (editors), Kluwer Academic Publishers, in press, Fall 2004 or Winter 2005.

10.2.4 PATENTS PENDING

1. W.D. Grover, J. Doucette, "Method for Design of Networks Based on *p*-Cycles," Provisional US and Canadian Patent filed by TRLabs September 2003, pending.
2. W.D. Grover, J. Doucette, "Topological Design of Survivable Mesh-Based Transport Networks," Provisional US Patent filed by TRLabs October 2001, pending.
3. W.D. Grover, J. Doucette, T. Bach, "Bi-Criterion Method of Designing Mesh-Restorable Networks," Provisional US Patent filed by Nortel November 2000, pending.
4. W.D. Grover, J. Doucette, "Restoration of Mesh Networks using Meta-Mesh of Chain Subnetworks," Provisional US Patent filed by TRLabs 25 October 2000, pending.

10.2.5 TECHNICAL REPORTS AND PRESENTATIONS

1. J. Doucette, W. D. Grover, "Node-Inclusive Span Restoration," *TRLabs Technology Forum 2004*, Saskatoon, SK, October 2004.
2. A. Grue, J. Doucette, W. D. Grover, "Characterizing the Performance of a Capacitated Iterative Design Algorithm for Fully Restorable *p*-Cycle Networks," TRLabs Technical Report TR-03-05, Edmonton, AB, December 2003.

3. J. Doucette, W. D. Grover, "Algorithmic Approaches for *p*-Cycle Network Design," *TRLabs Technology Forum 2003*, Calgary, AB, October 2003.
4. J. Doucette, W. D. Grover, "Maintenance Issues and SRLG Effects in the Design of Efficient Mesh-Based Restorable Networks," *TRLabs Technology Forum 2002*, Edmonton, AB, October 2002.
5. J. Doucette, W. D. Grover, W. Glenn, "MeshBuilder Prototype v1.2: A Toolkit for Design and Analysis of Mesh-Based Survivable Networks," *TRLabs Technology Forum 2002*, Edmonton, AB, October 2002.
6. W. Glenn, J. Doucette, W. D. Grover, "TRLabs *p*-Cycle Apriori Efficiency Metric: Report and Program Guide," TRLabs Technical Report TR-02-01, Edmonton, AB, May 2002.
7. J. Doucette, W. D. Grover, "Comparison of Mesh Protection and Restoration Schemes and the Dependency on Graph Connectivity," *TRLabs Technology Forum 2001*, Winnipeg, MB, October 2001.
8. J. Doucette, W. D. Grover, "An Effective Heuristic for Complete Mesh Network Design Including Basic Topology Optimization," *TRLabs Technology Forum 2000*, Saskatoon, SK, October 2000.
9. W. D. Grover, J. Doucette, "Design Method for a Meta-Mesh of Chain Subnetworks: Enhancing the Attractiveness of Mesh Networking on Low Connectivity Graphs," *TRLabs Technology Forum 2000*, Saskatoon, SK, October 2000.
10. J. Doucette, W. D. Grover, "Economy-of-Scale Effects on the Optimum Topology of Modular WDM Mesh-Restorable Networks," *TRLabs Technology Forum 1999*, Calgary, AB, October 1999.
11. J. Doucette, W. D. Grover, "Fundamental Effects of Fiber Route Topology on Mesh-Restorable Network Design," *TRLabs Technology Forum 1998*, Edmonton, AB, October 1998.

REFERENCES

[1] 360 Networks, "Network Map," accessed 18 May 2004, available on-line: http://www.360networks.com/images/Network_Map_EN.jpg, 17 February 2004.

[2] O. Aboul-Magd, et al., "Automatic Switched Optical Network (ASON) Architecture and Its Related Protocols," *IETF Internet Draft*, draft-ietf-ipo-ason-00.txt, July 2001.

[3] R. Ahuja, T. Magnanti, J. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Englewood Cliffs, New Jersey, Prentice Hall, 1993.

[4] R. Bhandari, *Survivable Networks: Algorithms for Diverse Routing*, Kluwer Academic Publishers, November 1998.

[5] R. Billinton, R. N. Allan, *Reliability Evaluation of Engineering Systems, 2nd Edition*, Plenum Press, New York and London, 1992.

[6] U. Black and S. Waters, *SONET and T1: Architectures for Digital Transport Networks*, 2nd edition, Upper Saddler River, New Jersey, Prentice Hall, 2002.

[7] R. R. Boorstyn and H. Frank, "Large-Scale Network Topological Optimization," *IEEE Transactions on Communications*, vol. 25, no. 1, pp. 29-47, January 1977.

[8] I. N. Bronshtein, K. A. Semendyayev, *Handbook of Mathematics*, 3rd Edition, English Language translation by K. A. Hirsch, New York, NY: Springer-Verlag, 1985.

[9] R. A. Brualdi, *Introductory Combinatorics*, 2nd Edition, Englewood Cliffs, New Jersey: Prentice-Hall, 1992.

[10] R. S. Cahn, *Wide Area Network Design: Concepts and Tools for Optimization*, Morgan Kaufman Publishers, San Francisco, CA, 1998.

[11] CANARIE, "About CA*net 4," accessed 8 November 2004, available on-line: http://www.canarie.ca/canet4/index.html, 2002.

[12] CANARIE, "CA*net 4 Outage Reports," accessed 8 November 2004, available on-line: http://dooka.canet4.net/network/report.php?type=Outage, November 2004.

[13] W. Chou, F. Ferrante, M. Balagangadhar, "Integrated Optimization of Distributed Processing Networks," *Proceedings of the AFIPS National Computer Conference*, pp. 795-811, 1978.

[14] T. Cinkler, T. Henk, G. Gordos, "Stochastic Algorithms for Design of Thrifty Single-Failure-Protected Networks," *Proceedings of the 2nd International Workshop on Design of Reliable Communication Networks (DRCN 2000)*, Munich, Germany, pp. 298-303, 9-12 April 2000.

[15] M. Clouqueur, W. D. Grover, "Computational and Design Studies on the Unavailability of Mesh-Restorable Networks," *Proceedings of the 2nd Interna-*

tional Workshop on Design of Reliable Communication Networks (DRCN 2000), Munich, Germany, pp. 181-186, April 2000.

[16] M. Clouqueur, W. D. Grover, "Availability Analysis of Span-Restorable Mesh Networks," *IEEE Journal on Selected Areas in Communications* (JSAC), *Special Issue on Recent Advances in Fundamentals of Network Management*, vol. 20, no. 4, pp. 810-821, May 2002.

[17] D. E. Comer, *Internetworking with TCP/IP: Principles, Protocols, and Architectures*, 4th edition, Prentice Hall, 2000.

[18] D. Crawford, "Fiber Optic Cable Dig-ups: Causes and Cures," *Network Reliability: A Report to the Nation – Compendium of Technical Papers*, National Engineering Consortium, Chicago, June 1993.

[19] F. R. B. Cruz, J. MacGregor Smith, G. R. Mateus, "Solving to Optimality the Uncapacitated Fixed-Charge Network Flow Problem," *Computers & Operations Research*, vol. 25, no. 1, pp. 67-81, January 1998.

[20] G. Dantzig, *Linear Programming and Extensions*, Princeton, New Jersey, Princeton University Press, 1963.

[21] B. Davie and Y. Rekhter, *MPLS: Technology and Applications*, Morgan Kaufmann, San Francisco, California, 2000.

[22] H. Dirilten and R. W. Donaldson, "Topological Design of Distributed Data Communications Networks Using Linear Regression Clustering," *IEEE Transactions on Communications*, vol. 25, no. 10, pp. 1083-1092, October 1977.

[23] J. Doucette, M. Clouqueur, W. D. Grover, "On the Availability and Capacity Requirements of Shared Backup Path-Protected Mesh Networks," *Optical Networks Magazine, Special Issue on Engineering the Next Generation Optical Internet*, vol. 4, no. 6, pp. 29-44, November/December 2003.

[24] J. Doucette, W. D. Grover, "Influence of Modularity and Economy-of-scale Effects on Design of Mesh-Restorable DWDM Networks," *IEEE Journal on Selected Areas in Communications (JSAC), Special Issue on Protocols and Architectures for Next Generation Optical WDM Networks*, vol. 18, no. 10, pp. 1912-1923, October 2000.

[25] J. Doucette, W. D. Grover, "Comparison of Mesh Protection and Restoration Schemes and the Dependency on Graph Connectivity," *Proceedings of 3rd International Workshop on Design of Reliable Communication Networks (DRCN 2001)*, Budapest, Hungary, pp. 121-128, October, 2001.

[26] J. Doucette, W. D. Grover, "Node-Inclusive Span Restoration in an Optical Mesh Transport Network," *Proceedings of the 19th Annual National Fiber Optic Engineers Conference (NFOEC 2003)*, Orlando, FL, in review, September 2003.

[27] J. Doucette, W. D. Grover, T. Bach, "Bi-criteria studies of mesh network restoration path-length versus capacity tradeoffs," *Proceedings of OSA Optical*

Fiber Communications Conference and Exhibit (OFC 2001), Anaheim, CA, pp. TuG2-1–TuG2-3, March 2001.

[28] R. D. Doverspike, B. Wilson, "Comparison of capacity efficiency of DCS network restoration routing techniques," *Journal of Network and Systems Management*, vol. 2, no. 2, pp. 95-123, 1994.

[29] M. Faloutsos, P. Faloutsos, C. Faloutsos, "On Power-Law Relationships of the Internet Topology," *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, Cambridge*, MA, pp. 251-262, August-September 1999.

[30] Federal Communication Commission, "Office of Engineering and Technology Web-site," accessed 02 April 2002, available on-line: http://www.fcc.gov/oet/, April 2002.

[31] Federal Communication Commission, "FCC Outage Report 02-026," *FCC Office of Engineering and Technology Outage Reports*, February 2002.

[32] Federal Communication Commission, "FCC Outage Report 02-028," *FCC Office of Engineering and Technology Outage Reports*, February 2002.

[33] Federal Communication Commission, "FCC Outage Report 02-034," *FCC Office of Engineering and Technology Outage Reports*, February 2002.

[34] B. Forouzan, *Introduction to Data Communications and Networking*, Boston, Massachusetts, McGraw-Hill, 1998.

[35] R. Fourer, D. M. Gay, B. W. Kernighan, *AMPL: A Modeling Language For Mathematical Programming*, 2nd edition, Pacific Grove, California, Brooks/Cole – Thompson Learning, 2003.

[36] H. Frank, I. T. Frisch, "Analysis and Design of Survivable Networks," *IEEE Transactions on Communications*, vol. 18, no. 5, pp. 29-47, October 1970.

[37] B. Gavish, K. Altinkemer, "Backbone network design tools with economic tradeoffs," *INFORMS Journal on Computing*, vol. 2, no. 3, pp. 236-252, Summer 1990.

[38] B. Gavish, P. Trudeau, M. Dror, M. Gendreau, L. Mason, "Fiberoptic Circuit Network Design Under Reliability Constraints," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 8, pp. 1181-1187, October 1989.

[39] B. Gendron, T. G. Crainic, A. Frangioni, "Multicommodity Capacitated Network Design," *Telecommunications Network Planning*, eds. B. Sanso, P. Soriano, Kluwer Academic Publishers, pp. 1-19, 1999.

[40] M. Gerla, H. Frank, W. Chou, J. Eckl, "A Cut Saturation Algorithm for Topological Design of Packet Switched Communication Networks," *Proceedings of the IEEE National Telecommunications Conference (NTC 1974)*, San Diego, CA, pp. 1074-1079, 2-4 December 1974.

[41] M. Gerla, L. Kleinrock, "On the Topological Design of Distributed Computer Networks," *IEEE Transactions on Communications*, vol. 25, no. 1, pp. 48-60, January 1977.

[42] M. Gerla, J. A. Suruagy Monteiro, R. Pazos, "Topology design and bandwidth allocation in ATM networks," *IEEE Journal on Selected Areas in Communications*, vol.7, no. 1253-1262, 1989.

[43] Global Crossing, "Global Crossing – Network Map," accessed 18 May 2004, available on-line: http://www.globalcrossing.com/xml/network/net_map.xml, 2004.

[44] Government of Canada, *The New National Dream: Networking the Nation for Broadband Access*, Report of the National Broadband Task Force, 18 June 2001.

[45] W. D. Grover, "The Selfhealing Network: A Fast Distributed Restoration Technique for Networks Using Digital Cross-connect Machines," *Proceedings of IEEE Global Telecommunications Conference (GlobeCom 1987)*, Toyko, Japan, pp. 1090-1095, November1987.

[46] W. D. Grover, "Distributed Restoration of the Transport Network," *Network Management into the 21st Century*, editors T. Plevyak, S. Aidarous, IEEE / IEE Press co-publication, ISBN 0-7803-1013-6, Chapter 11, pp. 337-417, February 1994.

[47] W. D. Grover, "Self-organizing Broad-band Transport Networks," *Proceedings of the IEEE*, vol. 85, no.10, pp. 1582-1611, October 1997.

[48] W. D. Grover, *Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET, and ATM Networking*, Prentice Hall PTR, Upper Saddle River, NJ, 2004.

[49] W. D. Grover, "The Protected Working Capacity Envelope Concept: An Alternate Paradigm for Automated Service Provisioning," *IEEE Communications Magazine (Special Issue on Management of Optical Networks and Services)*, vol. 42, no. 1, pp. 62-69, January 2004.

[50] W. D. Grover, T. D. Bilodeau, B. D. Venables, "Near Optimal Spare Capacity Planning in a Mesh Restorable Network," *Proceedings of IEEE Global Telecommunications Conference (GlobeCom 1991)*, vol. 3, pp. 2007-2012, December 1991.

[51] W. D. Grover, J. Doucette, "Topological design of span-restorable mesh transport networks," *Annals of Operations Research, Special Issue on Topological Design of Telecommunication Networks*, vol. 106, pp. 79-125, September 2001.

[52] W. D. Grover, J. Doucette, "Increasing the Efficiency of Span-Restorable Mesh Networks on Low-Connectivity Graphs," *Proceedings of the 3rd International Workshop on Design of Reliable Communication Networks (DRCN 2001)*, Budapest, Hungary, pp. 99-106, October, 2001.

[53] W. D. Grover, J. Doucette, "A Novel Heuristic for Topology Planning and Evolution of Optical Mesh Networks," *Proceedings of IEEE Global Telecommunications Conference (GlobeCom 2001)*, San Antonio, TX, pp. 2169-2173, November 2001.

[54] W. D. Grover, J. Doucette, "Design of a Meta-Mesh of Chain Sub-Networks: Enhancing the Attractiveness of Mesh-Restorable WDM Networking on Low Connectivity Graphs," *IEEE Journal on Selected Areas in Communications (JSAC), Special Issue on WDM-based Network Architectures*, vol. 20, no. 1, pp. 47-61, January 2002.

[55] W. D. Grover, V. Rawat, M. H. MacGregor, "Fast Heuristic Principle for Spare Capacity Placement in Mesh-Restorable SONET/SDH Transport Networks," *IEE Electronics Letters*, vol. 33, no. 3, pp. 195-196, 30 January 1997.

[56] W. D. Grover, D. Stamatelakis, "Cycle-Oriented Distributed Preconfiguration: Ring-like Speed with Mesh-like Capacity for Self-planning Network Restoration," *Proceedings of IEEE International Conference on Communications* (ICC 1998), Atlanta, GA, pp. 537-543, June 1998.

[57] R. Gupta, E. Chi, J. Walrand, "Different Algorithms for Normal and Protection Paths," *Proceedings of the 4th International Workshop on Design of Reliable Communication Networks (DRCN 2003)*, Banff, AB, Canada, pp. 189-196, 19-22 October 2003.

[58] P. E. Heegaard, B. E. Helvik, N. Stol, "Genetic algorithms for dimensioning of full service access networks," *Proceedings of the 14th Nordic Teletraffic Seminar (NTS-14)*, Copenhagen, Denmark, 18-20 August 1998.

[59] M. Herzberg, and S. J. Bye, "An optimal spare-capacity assignment model for survivable networks with hop limits," *Proceedings of IEEE Global Communications Conference (GlobeCom 1994)*, pp. 1601-1607, San Francisco, CA, December 1994.

[60] M. Herzberg, S. J. Bye, A. Utano, "The Hop-Limit Approach for Spare-Capacity Assignment in Survivable Networks," *IEEE/ACM Transactions on Networking*, vol. 3, no. 6, pp. 775-784, December 1995.

[61] ILOG, "ILOG CPLEX," accessed 18 May 2004, available online: http://www.ilog.com/products/cplex/, 2004.

[62] R. R. Iraschko, W. D. Grover, "A Highly Efficient Path-Restoration Protocol for Management of Optical Network Transport Integrity," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 5, pp. 779-793, May 2000.

[63] R. R. Iraschko, M. H. MacGregor, W. D. Grover, "Optimal Capacity Placement for Path Restoration in STM or ATM Mesh-Survivable Networks," *IEEE/ACM Transactions on Networking*, vol. 6, no. 3, pp. 325-336, June 1998.

[64] R. A. Johnson, *Miller & Freund's Probability & Statistics for Engineers*, 5th Edition, Prentice Hall, Englewood Cliffs, NJ, 1994.

[65] V. C. Jones, *High Availability Networking with Cisco*, Addison Wesley, Upper Saddle River, NJ, 2001.

[66] B. G. Józsa, D. Orincsay, "Shared Backup Path Optimization in Telecommunication Networks," *Proceedings of the 3rd International Workshop on Design*

of Reliable Communication Networks (DRCN 2001), Budapest, Hungary, pp. 251-257, October, 2001.

[67] J. M. Kahn, K.-P. Ho, "A Bottleneck for Optical Fibres," *Nature*, vol. 411, pp. 1007-1010, June 28, 2001.

[68] S. V. Kartalopoulos, *Introduction to DWDM Technology: Data in a Rainbow*, IEEE Press, Piscataway, NJ, 2000.

[69] R. Kawamura, K. Sato, I. Tokizawa, "Self-healing ATM networks based on virtual path concept," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 12, no. 1, pp. 120-127, January 1994.

[70] J. L. Kennington, M. W. Lewis, "The Path Restoration Version of the Spare Capacity Allocation Problem with Modularity Restrictions: Models, Algorithms, and an Empirical Analysis," Technical Report 98-CSE-13, Department of Computer Science And Engineering, Southern Methodist University, Dallas, December 1998.

[71] A. Kershenbaum, *Telecommunications Network Design Algorithms*, McGraw-Hill, New York, NY, 1993.

[72] A. Kershenbaum, P. Kermani, G. A. Grover, "MENTOR: An algorithm for mesh network topological optimization and routing," *IEEE Transactions on Communications*, vol. 39, no. 04, pp. 503-513, April 1991.

[73] H.-J. Kim, J. N. Hooker, "Solving Fixed-Charge Network Flow Problems with a Hybrid Optimization and Constraint Programming Approach," *Annals of Operations Research*, vol. 115, pp. 95-124, September 2002.

[74] S. Kini, M. Kodialam, T. V. Laksham, S. Sengupta, C. Villamizar, "Shared backup Label Switched Path restoration," *IETF Internet Draft*, draft-kini-restoration-shared-backup-01.txt, work in progress, May 2001.

[75] Level 3 Communications, "Network Maps," accessed 07 May 2004, available on-line: http://www.level3.com/577.html, 2004.

[76] M. Liotine, *Mission Critical Network Planning*, Artech House, Norwood, MA, 2003.

[77] Y. Liu, D. Tipper, "Spare Capacity Allocation for Non-Linear Link Cost and Failure-Dependent Path Restoration," *Proceedings of the 3rd International Workshop on Design of Reliable Communication Networks (DRCN 2001)*, Budapest, Hungary, pp. 243-250, October, 2001.

[78] Y. Liu, D. Tipper, P. Siripongwutikorn, "Approximating Optimal Spare Capacity Allocation by Successive Survivable Routing," *Proceedings of the 20th IEEE International Conference on Computer Communication (IEEE INFOCOM 2001)*, Anchorage, AK, pp.699-708, 24-28 April 2001.

[79] Lucent Technologies, Bell Labs Innovations, "Lucent – Press Release: WaveStar LambdaRouter," accessed 16 March 2004, available on-line: http://www.lucent.com/pressroom/lambda.html, 2004.

[80] I. J. Lustig, "Constraint Programming and its Relationship to Mathematical Programming," ILOG Corporation, Gentilly, France, December 2000.

[81] M. W. Maeda, "Management and Control of Transparent Optical Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 16, no. 7, pp. 1005-1023, September 1998.

[82] K. Maruyama, "Designing reliable packet switched networks," *Proceedings of IEEE International Conference on Communications (ICC 1978)*, Toronto, ON, pp. 493-498, June 1978.

[83] D. Medhi, D. Tipper, "Some Approaches to Solving a Multihour Broadband Network Capacity Design Problem with Single-Path Routing," *Telecommunication Systems*, vol. 13, no. 2, pp. 269-291, 2000.

[84] E. Modiano, P.J. Lin, "Traffic Grooming in WDM Networks," *IEEE Communications Magazine*, vol. 39, no. 7, pp. 124-129, July 2001.

[85] D. C. Montgomery, G. C. Runger, *Applied Statistics and Probability for Engineers*, 2nd Edition, John Wiley & Sons, New York, NY, 1999.

[86] G. D. Morley, *Analysis and Design of Ring-Based Transport Networks*, University of Alberta Ph.D. Thesis, February 2001.

[87] B. Mukherjee, "WDM Optical Communication Networks: Progress and Challenges," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, pp. 1810-1824, October 2000.

[88] Natural Sciences and Engineering Research Council of Canada, "Joint Infrastructure Interdependencies Research Program (JIIRP)," accessed 8 November 2004, available on-line: http://www.nserc.gc.ca/programs/jiirp_e.htm, March 2004.

[89] C. Nebo (née Ezema), *Topology Design of Mesh-Restorable Networks*, M.Sc. thesis, University of Alberta, Edmonton, AB, Canada, Spring 2003.

[90] OPNET, "OPNET Modeler: Accelerating Networking R&D," accessed 8 November 2004, available on-line: http://www.opnet.com/products/modeler/, 2004.

[91] OPNET, "Flow Analysis," accessed 18 May 2004, available on-line: http://www.opnet.com/products/modules/flow_analysis.html, 2002.

[92] M. Pickavet, P. Demeester, "Long-Term Planning of WDM Networks: A Comparison between Single-Period and Multi-Period Techniques," *Photonic Network Communications*, vol. 1, no. 4, pp. 331-346, December 1999.

[93] M. Pickavet, P. Demeester, "A Zoom-In approach to design SDH mesh-restorable networks," *Journal of Heuristics Special Edition on Heuristic Approaches for Telecommunications Network Management, Planning and Expansion*, vol. 6, no. 1, pp. 103- 126, April 2000.

[94] B. Ramamurthy, B. Mukherjee, "Wavelength Conversion in WDM Networking," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 16, no. 7, pp. 1061-1073, September 1998.

[95] R. Ramaswami, K. N. Sivarajan, "Routing and Wavelength Assignment in All-Optical Networks," *IEEE/ACM Transactions on Networking*, vol. 3, no. 5, pp. 489-500, October 1995.

[96] R. Ramaswami, K. N. Sivarajan, *Optical Networks: A Practical Perspective*, 2nd edition, Morgan Kaufmann Publishers, San Francisco, CA, 2002.

[97] I. Ross, Lloydminster Meridian Booster, "Cut Cable Kills Phone Service for Thousands," *Lloydminster Meridian Booster*, accessed 10 November 2004, available on-line: http://www.meridianbooster.com/story.php?id=126475, 10 November 2004.

[98] I. Rubin, "On Reliable Topological Structures for Message-Switching Communication Networks," *IEEE Transactions on Communications*, vol. 26, no. 1, pp. 62-74, January 1978.

[99] K. Ryan, "Providing efficient, scalable access networks," *Lightwave Magazine*, vol. 18, no. 6, June 2001.

[100] H. Sakauchi, Y. Nishimura, S. Hasegawa, "A Self-Healing Network with an Economical Spare Channel Assignment," *Proceedings of IEEE Global Telecommunications Conference (GlobeCom 1990)*, San Diego, CA, vol. 1, pp. 438-441, December 1990.

[101] C. V. Saradhi, C. S. Ram Murthy, "Dynamic Establishment of Segmented Protection Paths in Single and Multi-Fiber WDM Mesh Networks," *Optical Networking and Communications Conference (OptiComm 2002)*, Boston, MA, pp. 211-222, July-August 2002.

[102] M. Scheffel, *Configuration of p-Cycles in Optical Networks with Partial Wavelength Conversion*, Diploma thesis, Technische Universität München, Munich, Germany, Summer 2002.

[103] D. A. Schupke, C. G. Gruber, A. Autenrieth, "Optimal Configuration of *p*-Cycles in WDM Networks," *Proceedings of IEEE International Conference on Communications (ICC 2002)*, New York, NY, pp. 2761-2765, April-May 2002.

[104] D. A. Schupke, "Lower Bounds on the Spare Capacity for Link Protection and Link Restoration," *IEEE Communications Letters*, in press, 2004.

[105] S. Sengupta, R. Ramamurthy, "Capacity Efficient Distributed Routing of Mesh-Restored Lightpaths in Optical Networks," *Proceedings of IEEE Global Telecommunications Conference* (GlobeCom 2001), San Antonio, TX, pp. 2129-2133, November 2001.

[106] P. Soriano, C. Wynants, R. Seguin, M. Labbe, M. Gendreau, B. Fortz, "Design and dimensioning of survivable SDH / SONET Networks," *Telecommunications Network Planning*, eds. B. Sanso, P. Soriano, Kluwer Academic, pp. 148-167, 1999.

[107] D. Stamatelakis, W.D. Grover, "Theoretical Underpinnings for the Efficiency of Restorable Networks Using Pre-configured Cycles ("*p*-cycles")," *IEEE Transactions on Communications*, vol.48, no.8, pp. 1262-1265, August 2000.

[108] D. Stamatelakis, W. D. Grover, "IP Layer Restoration and Network Planning Based on Virtual Protection Cycles," *IEEE Journal on Selected Areas in Communications, Special Issue on Protocols and Architectures for Next Generation Optical WDM Networks*, vol.18, no.10, pp. 1938-1949, October, 2000.

[109] K. Steiglitz, P. Weiner, D. J. Kleitman, "The Design of Minimum-Cost Survivable Networks," *IEEE Transactions on Circuit Theory*, vol. 16, no. 4, pp. 455-460, November 1969.

[110] T. E. Stern, K. Bala, *Multiwavelength Optical Networks: A Layered Approach*, Prentice Hall, Upper Saddle River, NJ, 2000.

[111] P. Tomsu, C. Schmutzer, *Next Generation Optical Networks: The Convergence of IP Intelligence and Optical Technologies*, Prentice Hall, Upper Saddle River, NJ, 2002.

[112] TRLabs, "TRnet – Network of Distinction," *TRBits*, accessed 8 November 2004, available on-line: http://www.trlabs.ca/new/trbits/indexjan2004.html, January 2004.

[113] B. D. Venables, *Algorithms for the Spare Capacity Design of Mesh Restorable Networks*, M.Sc. Thesis, University of Alberta, Fall 1992.

[114] B. D. Venables, W. Grover, M.H. MacGregor, "Two Strategies for Spare Capacity Placement (SCP) in Mesh Restorable Networks," *Proceedings of IEEE International Conference on Communications (ICC 1993)*, pp. 267-271, Geneva, Switzerland, May 1993.

[115] A. J. Vernon, J. D. Portier, "Protection of Optical Channels in All-Optical Networks," Proceedings of the 18th Annual National Fiber Optic Engineers Conference (NFOEC 2002), pp. 1695-1706, Dallas, TX, September 2002.

[116] H. M. Wadsworth, ed., *Handbook of Statistical Methods for Engineers and Scientists*, McGraw-Hill Publishing Company, New York, NY, 1990.

[117] Y. Wang, *Modelling and solving single and multiple facility restoration problems*, Ph.D. dissertation, Sloan School of Management, Massachusetts Institute of Technology, pp. 32–33, June 1998.

[118] WilTel Communications, "Map of WilTel's Global Assets," accessed 18 May 2004, available on-line: http://www.wiltel.com/map/, February 2003.

[119] W. L. Winston, *Operations Research Applications and Algorithms*, 3rd Edition, Duxbury Press, Belmont, CA, 1994.

[120] T. S. Wu, *Fiber Network Service Survivability*, Artech House, 1992.

[121] Y. Xiong, L. G. Mason, "Restoration strategies and spare capacity requirements in self-healing ATM networks," *IEEE/ACM Transactions on Networking*, vol. 7, no. 1, pp. 98-110, February 1999.

[122] Y. Zheng, W. D. Grover, M. H. MacGregor, "Broadband Network Design with Controlled Exploitation of Flow-Convergence Overloads in ATM VP-Based Restoration," *Proceedings of the Canadian Conference on Broadband Research (CCBR 1997)*, Ottawa, ON, pp. 172-183, April 1997.

[123] G. K. Zipf, *Human Behavior and Principle of Least Effort: An Introduction to Human Ecology*, Addison Wesley, Cambridge, MA, 1949.

APPENDIX A
NETWORK TOPOLOGY FILES

A.1 15N30S1 MASTER NETWORK

```
NAME: 15n30s1
DATE LAST MODIFIED: Wednesday, July 25, 2001 2:51:30 PM MDT
MODIFIED BY: MeshBuilder(c)

NODE    X     Y      SIZE
N01    125   140      3
N02    268    55      4
N03    413   162      4
N04    525    97      3
N05    290   233      6
N06    612   218      5
N07    508   349      4
N08    572   485      4
N09    402   456      4
N10    259   325      6
N11    291   598      3
N12    292   483      4
N13    186   458      3
N14     49   421      4
N15     75   274      3

SPAN    O     D     LENGTH
S01    N01   N02   166.355
S02    N01   N05   189.404
S03    N01   N15   143.024
S04    N02   N03   180.205
S05    N02   N04   260.409
S06    N02   N05   179.354
S07    N03   N04   129.495
S08    N03   N05   142.021
S09    N03   N06   206.729
S10    N04   N06   149.030
S11    N05   N07   246.941
S12    N05   N10    97.082
S13    N05   N15   218.874
S14    N06   N07   167.263
S15    N07   N08   150.306
S16    N07   N09   150.615
S17    N08   N06   269.980
S18    N08   N11   302.870
S19    N09   N08   172.456
S20    N09   N10   193.933
S21    N10   N06   368.860
S22    N10   N14   230.903
S23    N11   N12   115.004
S24    N11   N14   299.822
S25    N12   N09   113.265
S26    N12   N10   161.409
S27    N13   N10   151.717
S28    N13   N12   108.908
S29    N14   N13   141.908
```

S30 N15 N14 149.282

A.1.1 15N30S1 NETWORK FAMILY MEMBERS' SPAN LISTINGS

15n30s1:
S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 S20 S21 **S22** S23 S24 S25 S26 S27 S28 S29 S30

15n30s1-29s:
S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 **S12** S13 S14 S15 S16
S17 S18 S19 S20 S21 S23 S24 S25 S26 S27 S28 S29 S30

15n30s1-28s:
S01 S02 S03 S04 S05 S06 S07 **S08** S09 S10 S11 S13 S14 S15 S16 S17
S18 S19 S20 S21 S23 S24 S25 S26 S27 S28 S29 S30

15n30s1-27s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S13 **S14** S15 S16 S17 S18
S19 S20 S21 S23 S24 S25 S26 S27 S28 S29 S30

15n30s1-26s:
S01 S02 S03 S04 S05 **S06** S07 S09 S10 S11 S13 S15 S16 S17 S18 S19
S20 S21 S23 S24 S25 S26 S27 S28 S29 S30

15n30s1-25s:
S01 S02 S03 S04 S05 S07 S09 S10 S11 S13 **S15** S16 S17 S18 S19 S20
S21 S23 S24 S25 S26 S27 S28 S29 S30

15n30s1-24s:
S01 S02 S03 S04 **S05** S07 S09 S10 S11 S13 S16 S17 S18 S19 S20 S21
S23 S24 S25 S26 S27 S28 S29 S30

15n30s1-23s:
S01 S02 S03 S04 S07 **S09** S10 S11 S13 S16 S17 S18 S19 S20 S21 S23
S24 S25 S26 S27 S28 S29 S30

15n30s1-22s:
S01 S02 S03 S04 S07 S10 S11 S13 S16 S17 S18 S19 S20 S21 S23 **S24**
S25 S26 S27 S28 S29 S30

15n30s1-21s:
S01 S02 S03 S04 S07 S10 S11 S13 S16 S17 S18 **S19** S20 S21 S23 S25
S26 S27 S28 S29 S30

15n30s1-20s:
S01 S02 S03 S04 S07 S10 S11 S13 S16 S17 S18 S20 S21 S23 S25 S26
S27 **S28** S29 S30

15n30s1-19s:
S01 S02 S03 S04 S07 S10 S11 S13 S16 S17 S18 S20 S21 S23 S25 **S26**
S27 S29 S30

15n30s1-18s:
S01 **S02** S03 S04 S07 S10 S11 S13 S16 S17 S18 S20 S21 S23 S25 S27
S29 S30

15n30s1-17s:
S01 S03 S04 S07 S10 S11 S13 S16 S17 S18 S20 **S21** S23 S25 S27 S29
S30

15n30s1-16s:
S01 S03 S04 S07 S10 S11 S13 S16 S17 S18 S20 S23 S25 S27 S29 S30

A.2 20N40S1 NETWORK FAMILY

```
NAME: 20n40s1
DATE LAST MODIFIED: Wednesday, July 25, 2001 2:52:03 PM MDT
MODIFIED BY: MeshBuilder(c)

NODE   X     Y      SIZE
N01    183   456    3
N02    222   322    3
N03    275   163    4
N04    266   297    5
N05    403   116    4
N06    470   253    4
N07    378   241    5
N08    331   314    4
N09    476   318    5
N10    607   311    3
N11    533   410    5
N12    634   482    3
N13    513   516    4
N14    406   389    6
N15    285   437    4
N16    338   473    5
N17    433   538    3
N18    510   637    3
N19    380   549    4
N20    260   543    3

SPAN   O     D     LENGTH
S01    N01   N02   139.560
S02    N01   N04   179.360
S03    N01   N20   116.181
S04    N02   N03   167.601
S05    N02   N04    50.606
S06    N03   N07   129.201
S07    N04   N03   134.302
S08    N04   N05   227.002
S09    N04   N08    67.186
S10    N05   N03   136.356
S11    N05   N06   152.506
S12    N05   N07   127.475
S13    N06   N10   148.772
S14    N06   N15   260.923
S15    N07   N06    92.779
S16    N07   N08    86.822
S17    N07   N09   124.631
S18    N08   N09   145.055
S19    N08   N15   131.320
S20    N09   N11   108.227
S21    N09   N14    99.705
S22    N10   N09   131.187
S23    N10   N12   173.118
S24    N11   N12   124.036
S25    N11   N13   107.870
S26    N12   N18   198.497
S27    N13   N17    82.970
S28    N14   N11   128.725
S29    N14   N13   166.066
S30    N14   N17   151.427
```

```
S31   N15   N14  130.173
S32   N15   N16   64.070
S33   N16   N11  204.924
S34   N16   N14  108.074
S35   N16   N20  104.805
S36   N17   N19   54.129
S37   N18   N13  121.037
S38   N19   N16   86.833
S39   N19   N18  156.984
S40   N20   N19  120.150
```

A.2.1 20N40S1 NETWORK FAMILY MEMBERS' SPAN LISTINGS

20n40s1:
S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 S20 S21 S22 S23 S24 **S25** S26 S27 S28 S29 S30 S31 S32
S33 S34 S35 S36 S37 S38 S39 S40

20n40s1-39s:
S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 **S20** S21 S22 S23 S24 S26 S27 S28 S29 S30 S31 S32 S33
S34 S35 S36 S37 S38 S39 S40

20n40s1-38s:
S01 S02 S03 S04 S05 S06 S07 S08 **S09** S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 S21 S22 S23 S24 S26 S27 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 S38 S39 S40

20n40s1-37s:
S01 S02 S03 S04 S05 **S06** S07 S08 S10 S11 S12 S13 S14 S15 S16 S17
S18 S19 S21 S22 S23 S24 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35
S36 S37 S38 S39 S40

20n40s1-36s:
S01 S02 S03 S04 S05 S07 S08 S10 S11 S12 S13 S14 S15 S16 S17 S18
S19 S21 S22 S23 **S24** S26 S27 S28 S29 S30 S31 S32 S33 S34 S35 S36
S37 S38 S39 S40

20n40s1-35s:
S01 S02 S03 S04 S05 S07 **S08** S10 S11 S12 S13 S14 S15 S16 S17 S18
S19 S21 S22 S23 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35 S36 S37
S38 S39 S40

20n40s1-34s:
S01 S02 S03 S04 S05 S07 S10 **S11** S12 S13 S14 S15 S16 S17 S18 S19
S21 S22 S23 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35 S36 S37 S38
S39 S40

20n40s1-33s:
S01 S02 S03 S04 S05 S07 S10 S12 S13 S14 S15 S16 S17 S18 S19 S21
S22 S23 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35 S36 S37 S38 **S39**
S40

20n40s1-32s:
S01 S02 S03 S04 S05 S07 S10 S12 **S13** S14 S15 S16 S17 S18 S19 S21
S22 S23 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35 S36 S37 S38 S40

20n40s1-31s:
S01 S02 S03 S04 S05 S07 S10 S12 S14 S15 S16 S17 S18 S19 S21 S22
S23 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35 **S36** S37 S38 S40

20n40s1-30s:
S01 S02 S03 S04 S05 S07 S10 S12 S14 S15 S16 **S17** S18 S19 S21 S22
S23 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35 S37 S38 S40

20n40s1-29s:
S01 S02 S03 S04 S05 **S07** S10 S12 S14 S15 S16 S18 S19 S21 S22 S23
S26 S27 S28 S29 S30 S31 S32 S33 S34 S35 S37 S38 S40

20n40s1-28s:
S01 S02 S03 S04 S05 S10 S12 S14 S15 S16 S18 S19 S21 S22 S23 S26
S27 S28 **S29** S30 S31 S32 S33 S34 S35 S37 S38 S40

20n40s1-27s:
S01 S02 S03 S04 S05 S10 S12 S14 S15 S16 S18 S19 S21 S22 S23 S26
S27 S28 S30 S31 S32 S33 **S34** S35 S37 S38 S40

20n40s1-26s:
S01 S02 S03 S04 S05 S10 S12 S14 S15 S16 S18 **S19** S21 S22 S23 S26
S27 S28 S30 S31 S32 S33 S35 S37 S38 S40

20n40s1-25s:
S01 S02 S03 S04 S05 S10 S12 S14 S15 S16 S18 S21 S22 S23 S26 S27
S28 S30 **S31** S32 S33 S35 S37 S38 S40

20n40s1-24s:
S01 S02 S03 S04 S05 S10 S12 S14 S15 S16 S18 S21 S22 S23 S26 S27
S28 S30 S32 S33 S35 S37 S38 S40

20n40s1-23s:
S02 S03 S04 S05 S10 S12 S14 S15 S16 S18 S21 S22 S23 S26 S27 S28
S30 S32 S33 **S35** S37 S38 S40

20n40s1-22s:
S02 S03 S04 S05 S10 S12 S14 S15 S16 S18 **S21** S22 S23 S26 S27 S28
S30 S32 S33 S37 S38 S40

20n40s1-21s:
S02 S03 S04 S05 S10 S12 S14 S15 S16 S18 S22 S23 S26 S27 S28 S30
S32 S33 S37 S38 S40

A.3 25N50S1 NETWORK FAMILY

```
NAME: 25n50s1
DATE LAST MODIFIED: Wednesday, July 25, 2001 2:52:26 PM MDT
MODIFIED BY: MeshBuilder(c)

NODE    X      Y     SIZE
N01     92    136      3
N02    175     78      3
N03    266    117      3
N04    359     32      3
N05    390    159      3
N06    344    239      5
N07    480    223      4
N08    561    195      4
N09    515    297      3
N10    432    290      6
N11    564    411      5
N12    446    414      4
N13    504    482      3
N14    390    454      4
N15    351    316      7
N16    337    556      3
N17    168    571      4
N18    212    427      5
N19    127    451      4
N20    193    375      3
N21    155    283      6
N22     52    349      4
N23    105    254      3
N24    245    286      4
N25    210    190      4

SPAN    O      D     LENGTH
S01    N01    N02   101.257
S02    N01    N21   159.931
S03    N02    N03    99.005
S04    N02    N25   117.341
S05    N03    N04   125.992
S06    N03    N06   144.803
S07    N05    N04   130.729
S08    N05    N07   110.436
S09    N06    N24   109.590
S10    N07    N06   136.938
S11    N07    N08    85.703
S12    N07    N10    82.420
S13    N08    N04   259.563
S14    N08    N09   111.893
S15    N08    N10   160.206
S16    N09    N10    83.295
S17    N09    N11   124.085
S18    N10    N06   101.710
S19    N11    N10   179.067
S20    N11    N13    92.957
S21    N11    N15   233.225
S22    N12    N11   118.038
S23    N12    N13    89.376
S24    N13    N14   117.388
S25    N14    N12    68.819
```

S26	N15	N10	85.071
S27	N15	N12	136.488
S28	N15	N14	143.405
S29	N15	N17	313.869
S30	N16	N14	114.948
S31	N16	N18	179.627
S32	N17	N16	169.664
S33	N17	N19	126.811
S34	N18	N15	177.882
S35	N18	N17	150.572
S36	N19	N18	88.323
S37	N20	N18	55.362
S38	N21	N19	170.317
S39	N21	N22	122.332
S40	N21	N24	90.050
S41	N22	N19	126.606
S42	N22	N20	143.377
S43	N23	N01	118.714
S44	N23	N21	57.801
S45	N23	N22	108.784
S46	N24	N15	110.164
S47	N24	N20	103.078
S48	N25	N05	182.650
S49	N25	N06	142.678
S50	N25	N21	108.046

A.3.1 25N50S1 NETWORK FAMILY MEMBERS' SPAN LISTINGS

25n50s1:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32
S33 S34 **S35** S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48
S49 S50

25n50s1-49s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 **S28** S29 S30 S31 S32
S33 S34 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49
S50

25n50s1-48s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S29 S30 S31 S32 S33
S34 **S36** S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50

25n50s1-47s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 **S15** S16
S17 S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S29 S30 S31 S32 S33
S34 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50

25n50s1-46s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S16 S17
S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34
S37 **S38** S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50

25n50s1-45s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S16 S17
S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34
S37 S39 S40 S41 S42 S43 S44 **S45** S46 S47 S48 S49 S50

25n50s1-44s:
S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S16 S17
S18 S19 S20 S21 S22 **S23** S24 S25 S26 S27 S29 S30 S31 S32 S33 S34
S37 S39 S40 S41 S42 S43 S44 S46 S47 S48 S49 S50

25n50s1-43s:
S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S16 S17
S18 S19 S20 S21 S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 **S37**
S39 S40 S41 S42 S43 S44 S46 S47 S48 S49 S50

25n50s1-42s:
S01 S02 S03 S04 S05 S06 **S07** S08 S09 S10 S11 S12 S13 S14 S16 S17
S18 S19 S20 S21 S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 S39
S40 S41 S42 S43 S44 S46 S47 S48 S49 S50

25n50s1-41s:
S01 S02 S03 S04 S05 S06 S08 S09 S10 S11 S12 S13 S14 S16 S17 S18
S19 S20 S21 S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 S39 S40
S41 S42 S43 S44 **S46** S47 S48 S49 S50

25n50s1-40s:
S01 S02 S03 S04 S05 S06 S08 S09 S10 S11 S12 S13 S14 S16 S17 S18
S19 S20 S21 S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 **S39** S40
S41 S42 S43 S44 S47 S48 S49 S50

25n50s1-39s:
S01 S02 S03 S04 S05 S06 S08 S09 S10 S11 **S12** S13 S14 S16 S17 S18
S19 S20 S21 S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 S40 S41
S42 S43 S44 S47 S48 S49 S50

25n50s1-38s:
S01 S02 S03 S04 S05 S06 S08 S09 S10 S11 S13 S14 S16 S17 S18 S19
S20 S21 S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 S40 S41 S42
S43 S44 S47 S48 **S49** S50

25n50s1-37s:
S01 S02 **S03** S04 S05 S06 S08 S09 S10 S11 S13 S14 S16 S17 S18 S19
S20 S21 S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 S40 S41 S42
S43 S44 S47 S48 S50

25n50s1-36s:
S01 S02 S04 S05 S06 S08 S09 S10 **S11** S13 S14 S16 S17 S18 S19 S20
S21 S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 S40 S41 S42 S43
S44 S47 S48 S50

25n50s1-35s:
S01 S02 S04 S05 S06 S08 S09 S10 S13 S14 S16 S17 **S18** S19 S20 S21
S22 S24 S25 S26 S27 S29 S30 S31 S32 S33 S34 S40 S41 S42 S43 S44
S47 S48 S50

25n50s1-34s:
S01 S02 S04 S05 S06 S08 S09 S10 S13 S14 S16 S17 S19 S20 S21 S22
S24 S25 **S26** S27 S29 S30 S31 S32 S33 S34 S40 S41 S42 S43 S44 S47
S48 S50

25n50s1-33s:
S01 S02 S04 S05 S06 S08 S09 S10 S13 S14 S16 S17 S19 S20 S21 S22
S24 S25 S27 **S29** S30 S31 S32 S33 S34 S40 S41 S42 S43 S44 S47 S48
S50

25n50s1-32s:
S01 S02 S04 S05 S06 S08 S09 S10 S13 S14 S16 S17 S19 S20 S21 S22
S24 S25 S27 **S30** S31 S32 S33 S34 S40 S41 S42 S43 S44 S47 S48 S50

25n50s1-31s:
S01 S02 S04 S05 S06 S08 S09 S10 S13 S14 S16 **S17** S19 S20 S21 S22
S24 S25 S27 S31 S32 S33 S34 S40 S41 S42 S43 S44 S47 S48 S50

25n50s1-30s:
S01 S02 S04 S05 S06 S08 S09 S10 S13 S14 S16 S19 S20 S21 **S22** S24
S25 S27 S31 S32 S33 S34 S40 S41 S42 S43 S44 S47 S48 S50

25n50s1-29s:
S01 S02 S04 S05 S06 S08 S09 S10 S13 S14 S16 S19 S20 **S21** S24 S25
S27 S31 S32 S33 S34 S40 S41 S42 S43 S44 S47 S48 S50

25n50s1-28s:
S01 S02 S04 S05 S06 S08 **S09** S10 S13 S14 S16 S19 S20 S24 S25 S27
S31 S32 S33 S34 S40 S41 S42 S43 S44 S47 S48 S50

25n50s1-27s:
S01 S02 S04 S05 S06 S08 S10 S13 S14 S16 S19 S20 S24 S25 S27 S31
S32 S33 S34 S40 S41 S42 S43 S44 S47 S48 **S50**

25n50s1-26s:
S01 **S02** S04 S05 S06 S08 S10 S13 S14 S16 S19 S20 S24 S25 S27 S31
S32 S33 S34 S40 S41 S42 S43 S44 S47 S48

25n50s1-25s:
S01 S04 S05 S06 S08 S10 S13 S14 S16 S19 S20 S24 S25 S27 S31 S32
S33 S34 S40 S41 S42 S43 S44 S47 S48

A.4 30N60S1 NETWORK FAMILY

```
NAME: 30n60s1
DATE LAST MODIFIED: Wednesday, July 25, 2001 2:52:51 PM MDT
MODIFIED BY: MeshBuilder(c)

NODE    X      Y     SIZE
N01    155    169     4
N02    145     72     3
N03    206     40     3
N04    350     30     3
N05    350     76     5
N06    220    129     5
N07    322    134     4
N08    437    120     4
N09    493    153     4
N10    567    133     3
N11    613    191     3
N12    534    228     5
N13    451    223     6
N14    520    309     5
N15    658    331     4
N16    587    390     5
N17    466    422     3
N18    589    445     4
N19    664    475     3
N20    594    537     3
N21    412    533     4
N22    519    539     3
N23    320    428     3
N24    117    311     3
N25    270    339     4
N26    381    391     6
N27    370    201     5
N28    301    229     5
N29     72    227     3
N30    234    221     5

SPAN    O      D     LENGTH
S01    N01    N02    97.514
S02    N01    N30    94.578
S03    N03    N02    68.884
S04    N03    N07   149.305
S05    N04    N03   144.347
S06    N04    N06   163.404
S07    N05    N04    46.000
S08    N05    N07    64.405
S09    N05    N13   178.354
S10    N06    N01    76.322
S11    N06    N02    94.202
S12    N06    N05   140.389
S13    N07    N08   115.849
S14    N08    N09    65.000
S15    N08    N13   103.947
S16    N08    N27   105.119
S17    N09    N10    76.655
S18    N09    N11   125.873
S19    N09    N13    81.633
S20    N10    N05   224.361
```

```
S21   N11   N15  147.054
S22   N12   N10  100.568
S23   N12   N11   87.235
S24   N13   N12   83.150
S25   N14   N12   82.201
S26   N14   N26  161.385
S27   N15   N12  161.199
S28   N15   N16   92.315
S29   N16   N14  105.119
S30   N16   N19  114.691
S31   N17   N14  125.240
S32   N17   N16  125.160
S33   N17   N21  123.438
S34   N18   N16   55.036
S35   N18   N20   92.136
S36   N19   N15  144.125
S37   N20   N19   93.509
S38   N21   N18  197.669
S39   N22   N18  117.201
S40   N22   N20   75.027
S41   N22   N21  107.168
S42   N23   N21  139.603
S43   N24   N23  234.303
S44   N24   N25  155.541
S45   N25   N26  122.577
S46   N25   N30  123.369
S47   N26   N13  182.000
S48   N26   N23   71.344
S49   N26   N27  190.318
S50   N27   N07   82.420
S51   N27   N13   83.934
S52   N28   N14  233.154
S53   N28   N25  114.285
S54   N28   N26  180.677
S55   N28   N27   74.465
S56   N29   N01  101.257
S57   N29   N24   95.294
S58   N29   N30  162.111
S59   N30   N06   93.059
S60   N30   N28   67.476
```

A.4.1 30N60S1 NETWORK FAMILY MEMBERS' SPAN LISTINGS

30n60s1:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 **S16**
S17 S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32
S33 S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48
S49 S50 S51 S52 S53 S54 S55 S56 S57 S58 S59 S60

30n60s1-59s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17
S18 S19 S20 S21 S22 S23 S24 S25 S26 **S27** S28 S29 S30 S31 S32 S33
S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49
S50 S51 S52 S53 S54 S55 S56 S57 S58 S59 S60

30n60s1-58s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17
S18 S19 S20 S21 S22 S23 **S24** S25 S26 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50
S51 S52 S53 S54 S55 S56 S57 S58 S59 S60

30n60s1-57s:
S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17
S18 S19 S20 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S34 S35
S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 **S49** S50 S51
S52 S53 S54 S55 S56 S57 S58 S59 S60

30n60s1-56s:
S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17
S18 S19 S20 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S34 **S35**
S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S50 S51 S52
S53 S54 S55 S56 S57 S58 S59 S60

30n60s1-55s:
S01 S02 S03 S04 S05 S06 S07 **S08** S09 S10 S11 S12 S13 S14 S15 S17
S18 S19 S20 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S34 S36
S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S50 S51 S52 S53
S54 S55 S56 S57 S58 S59 S60

30n60s1-54s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 S15 S17 S18
S19 S20 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S34 S36 S37
S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S50 S51 S52 S53 **S54**
S55 S56 S57 S58 S59 S60

30n60s1-53s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 S15 S17 S18
S19 S20 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S34 **S36** S37
S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S50 S51 S52 S53 S55
S56 S57 S58 S59 S60

30n60s1-52s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 S15 S17 S18
S19 S20 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S34 S37 S38
S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S50 S51 **S52** S53 S55 S56
S57 S58 S59 S60

30n60s1-51s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 S15 S17 S18
S19 S20 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 **S34** S37 S38
S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S50 S51 S53 S55 S56 S57
S58 S59 S60

30n60s1-50s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 S15 S17 S18
S19 **S20** S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S37 S38 S39
S40 S41 S42 S43 S44 S45 S46 S47 S48 S50 S51 S53 S55 S56 S57 S58
S59 S60

30n60s1-49s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 S15 S17 S18
S19 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S37 S38 S39 S40
S41 S42 S43 S44 S45 S46 S47 S48 S50 **S51** S53 S55 S56 S57 S58 S59
S60

30n60s1-48s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 **S15** S17 S18
S19 S21 S22 S23 S25 S26 S28 S29 S30 S31 S32 S33 S37 S38 S39 S40
S41 S42 S43 S44 S45 S46 S47 S48 S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-47s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 S17 S18 S19
S21 S22 S23 S25 **S26** S28 S29 S30 S31 S32 S33 S37 S38 S39 S40 S41
S42 S43 S44 S45 S46 S47 S48 S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-46s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 S12 S13 S14 S17 S18 **S19**
S21 S22 S23 S25 S28 S29 S30 S31 S32 S33 S37 S38 S39 S40 S41 S42
S43 S44 S45 S46 S47 S48 S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-45s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 S11 **S12** S13 S14 S17 S18 S21
S22 S23 S25 S28 S29 S30 S31 S32 S33 S37 S38 S39 S40 S41 S42 S43
S44 S45 S46 S47 S48 S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-44s:
S01 S02 S03 S04 S05 S06 S07 S09 S10 **S11** S13 S14 S17 S18 S21 S22
S23 S25 S28 S29 S30 S31 S32 S33 S37 S38 S39 S40 S41 S42 S43 S44
S45 S46 S47 S48 S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-43s:
S01 S02 S03 S04 S05 **S06** S07 S09 S10 S13 S14 S17 S18 S21 S22 S23
S25 S28 S29 S30 S31 S32 S33 S37 S38 S39 S40 S41 S42 S43 S44 S45
S46 S47 S48 S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-42s:
S01 S02 S03 S04 S05 S07 S09 S10 S13 S14 S17 S18 S21 S22 S23 S25
S28 **S29** S30 S31 S32 S33 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46
S47 S48 S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-41s:
S01 S02 S03 **S04** S05 S07 S09 S10 S13 S14 S17 S18 S21 S22 S23 S25
S28 S30 S31 S32 S33 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47
S48 S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-40s:
S01 S02 S03 S05 S07 S09 S10 S13 S14 S17 **S18** S21 S22 S23 S25 S28
S30 S31 S32 S33 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48
S50 S53 S55 S56 S57 S58 S59 S60

30n60s1-39s:
S01 S02 S03 S05 S07 S09 S10 S13 S14 S17 S21 S22 S23 S25 S28 S30
S31 S32 **S33** S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S50
S53 S55 S56 S57 S58 S59 S60

30n60s1-38s:
S01 S02 S03 S05 S07 S09 S10 S13 S14 S17 S21 S22 S23 S25 S28 S30
S31 S32 S37 S38 S39 S40 S41 S42 S43 S44 **S45** S46 S47 S48 S50 S53
S55 S56 S57 S58 S59 S60

30n60s1-37s:
S01 S02 S03 S05 S07 S09 S10 S13 S14 S17 S21 S22 S23 S25 S28 S30
S31 S32 S37 S38 S39 S40 S41 S42 S43 S44 S46 S47 S48 S50 S53 S55
S56 S57 **S58** S59 S60

30n60s1-36s:
S01 S02 S03 S05 S07 S09 S10 S13 S14 S17 S21 S22 S23 S25 S28 S30
S31 S32 S37 S38 S39 S40 **S41** S42 S43 S44 S46 S47 S48 S50 S53 S55
S56 S57 S59 S60

30n60s1-35s:
S01 S02 S03 S05 S07 S09 S10 S13 S14 S17 S21 S22 S23 S25 S28 S30
S31 S32 S37 S38 S39 S40 S42 S43 S44 **S46** S47 S48 S50 S53 S55 S56
S57 S59 S60

30n60s1-34s:
S01 **S02** S03 S05 S07 S09 S10 S13 S14 S17 S21 S22 S23 S25 S28 S30
S31 S32 S37 S38 S39 S40 S42 S43 S44 S47 S48 S50 S53 S55 S56 S57
S59 S60

30n60s1-33s:

S01 S03 S05 S07 S09 S10 S13 S14 S17 S21 S22 S23 S25 S28 S30 S31
S32 S37 S38 S39 S40 S42 **S43** S44 S47 S48 S50 S53 S55 S56 S57 S59
S60

30n60s1-32s:

S01 S03 S05 S07 S09 S10 S13 S14 S17 S21 S22 S23 S25 S28 S30 S31
S32 S37 S38 S39 S40 S42 S44 S47 S48 S50 S53 S55 S56 S57 S59 S60

A.5 35N70S1 NETWORK FAMILY

NAME: 35n70s1
DATE LAST MODIFIED: Wednesday, July 25, 2001 2:54:32 PM MDT
MODIFIED BY: MeshBuilder(c)

NODE	**X**	**Y**	**SIZE**
N01	370	137	5
N02	591	158	3
N03	524	224	5
N04	410	215	4
N05	434	288	3
N06	674	225	3
N07	418	330	3
N08	581	286	6
N09	635	332	3
N10	451	362	3
N11	582	381	4
N12	669	451	3
N13	578	489	7
N14	492	420	5
N15	477	489	4
N16	671	615	3
N17	623	651	3
N18	558	568	5
N19	513	617	3
N20	433	683	3
N21	405	585	6
N22	424	445	5
N23	308	572	5
N24	277	657	4
N25	169	607	4
N26	191	488	5
N27	252	395	6
N28	286	466	4
N29	367	472	4
N30	200	331	3
N31	114	454	3
N32	263	158	3
N33	361	397	4
N34	291	340	3
N35	354	252	3

SPAN	**O**	**D**	**LENGTH**
S01	N01	N02	221.995
S02	N01	N04	87.658
S03	N02	N03	94.048
S04	N02	N06	106.668
S05	N03	N01	176.876
S06	N03	N08	84.220
S07	N04	N03	114.355
S08	N05	N03	110.436
S09	N05	N04	76.844
S10	N06	N08	111.221
S11	N07	N05	44.944
S12	N07	N08	168.834
S13	N08	N10	150.586
S14	N08	N11	95.005
S15	N09	N06	113.886

```
S16   N09   N08    70.937
S17   N09   N12   123.762
S18   N10   N14    71.028
S19   N11   N10   132.371
S20   N11   N13   108.074
S21   N11   N14    98.087
S22   N12   N13    98.615
S23   N13   N16   156.605
S24   N13   N18    81.492
S25   N13   N21   197.851
S26   N14   N13   110.259
S27   N14   N15    70.612
S28   N15   N13   101.000
S29   N15   N22    68.884
S30   N16   N12   164.012
S31   N16   N17    60.000
S32   N17   N18   105.423
S33   N17   N19   115.135
S34   N18   N15   113.146
S35   N19   N18    66.528
S36   N19   N20   103.711
S37   N20   N24   158.152
S38   N21   N18   153.942
S39   N21   N20   101.922
S40   N21   N23    97.867
S41   N22   N14    72.450
S42   N22   N21   141.283
S43   N22   N29    63.071
S44   N23   N28   108.259
S45   N24   N23    90.477
S46   N24   N26   189.623
S47   N25   N23   143.339
S48   N25   N24   119.013
S49   N25   N31   162.585
S50   N26   N25   121.017
S51   N26   N28    97.514
S52   N26   N31    84.172
S53   N27   N26   111.221
S54   N27   N30    82.462
S55   N27   N32   237.255
S56   N28   N27    78.721
S57   N28   N29    81.222
S58   N29   N21   119.218
S59   N29   N23   116.108
S60   N30   N31   150.083
S61   N32   N01   109.041
S62   N32   N30   184.114
S63   N33   N07    87.966
S64   N33   N22    79.202
S65   N33   N27   109.018
S66   N33   N34    90.272
S67   N34   N27    67.424
S68   N35   N01   116.108
S69   N35   N04    67.119
S70   N35   N34   108.227
```

A.5.1 35N70S1 NETWORK FAMILY MEMBERS' SPAN LISTINGS

35n70s1:

S01 **S02** S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32
S33 S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48
S49 S50 S51 S52 S53 S54 S55 S56 S57 S58 S59 S60 S61 S62 S63 S64
S65 S66 S67 S68 S69 S70

35n70s1-69s:

S01 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16 S17
S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33
S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 **S48** S49
S50 S51 S52 S53 S54 S55 S56 S57 S58 S59 S60 S61 S62 S63 S64 S65
S66 S67 S68 S69 S70

35n70s1-68s:

S01 S03 S04 **S05** S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16 S17
S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33
S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S49 S50
S51 S52 S53 S54 S55 S56 S57 S58 S59 S60 S61 S62 S63 S64 S65 S66
S67 S68 S69 S70

35n70s1-67s:

S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16 S17 S18
S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S49 S50 S51
S52 **S53** S54 S55 S56 S57 S58 S59 S60 S61 S62 S63 S64 S65 S66 S67
S68 S69 S70

35n70s1-66s:

S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16 S17 S18
S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 S38 S39 **S40** S41 S42 S43 S44 S45 S46 S47 S49 S50 S51
S52 S54 S55 S56 S57 S58 S59 S60 S61 S62 S63 S64 S65 S66 S67 S68
S69 S70

35n70s1-65s:

S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16 S17 S18
S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 S38 S39 S41 S42 S43 S44 S45 S46 S47 S49 S50 S51 S52
S54 S55 S56 S57 S58 **S59** S60 S61 S62 S63 S64 S65 S66 S67 S68 S69
S70

35n70s1-64s:

S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16 S17 S18
S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 **S38** S39 S41 S42 S43 S44 S45 S46 S47 S49 S50 S51 S52
S54 S55 S56 S57 S58 S60 S61 S62 S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-63s:

S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16 S17 S18
S19 S20 S21 S22 S23 **S24** S25 S26 S27 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 S39 S41 S42 S43 S44 S45 S46 S47 S49 S50 S51 S52 S54
S55 S56 S57 S58 S60 S61 S62 S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-62s:

S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 **S14** S15 S16 S17 S18
S19 S20 S21 S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35
S36 S37 S39 S41 S42 S43 S44 S45 S46 S47 S49 S50 S51 S52 S54 S55
S56 S57 S58 S60 S61 S62 S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-61s:
S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S15 S16 S17 S18 S19
S20 S21 S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 **S34** S35 S36
S37 S39 S41 S42 S43 S44 S45 S46 S47 S49 S50 S51 S52 S54 S55 S56
S57 S58 S60 S61 S62 S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-60s:
S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S15 S16 S17 S18 S19
S20 S21 S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 S35 S36 S37
S39 S41 S42 S43 S44 **S45** S46 S47 S49 S50 S51 S52 S54 S55 S56 S57
S58 S60 S61 S62 S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-59s:
S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S15 S16 **S17** S18 S19
S20 S21 S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 S35 S36 S37
S39 S41 S42 S43 S44 S46 S47 S49 S50 S51 S52 S54 S55 S56 S57 S58
S60 S61 S62 S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-58s:
S01 S03 S04 S06 S07 S08 S09 S10 S11 S12 S13 S15 S16 S18 S19 S20
S21 S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 S35 S36 S37 S39
S41 S42 S43 S44 S46 S47 **S49** S50 S51 S52 S54 S55 S56 S57 S58 S60
S61 S62 S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-57s:
S01 S03 S04 S06 S07 S08 S09 S10 S11 **S12** S13 S15 S16 S18 S19 S20
S21 S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 S35 S36 S37 S39
S41 S42 S43 S44 S46 S47 S50 S51 S52 S54 S55 S56 S57 S58 S60 S61
S62 S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-56s:
S01 S03 S04 S06 S07 S08 S09 S10 S11 S13 S15 S16 S18 S19 S20 S21
S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 S35 S36 S37 S39 S41
S42 S43 S44 S46 S47 S50 S51 S52 S54 S55 S56 **S57** S58 S60 S61 S62
S63 S64 S65 S66 S67 S68 S69 S70

35n70s1-55s:
S01 S03 S04 S06 S07 S08 S09 S10 S11 S13 S15 S16 S18 S19 S20 S21
S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 S35 S36 S37 S39 S41
S42 S43 S44 S46 S47 S50 S51 S52 S54 S55 S56 S58 S60 S61 S62 S63
S64 S65 S66 S67 **S68** S69 S70

35n70s1-54s:
S01 S03 S04 **S06** S07 S08 S09 S10 S11 S13 S15 S16 S18 S19 S20 S21
S22 S23 S25 S26 S27 S28 S29 S30 S31 S32 S33 S35 S36 S37 S39 S41
S42 S43 S44 S46 S47 S50 S51 S52 S54 S55 S56 S58 S60 S61 S62 S63
S64 S65 S66 S67 S69 S70

35n70s1-53s:
S01 S03 S04 S07 S08 S09 S10 S11 S13 S15 S16 S18 S19 S20 S21 S22
S23 S25 S26 S27 S28 S29 S30 S31 S32 **S33** S35 S36 S37 S39 S41 S42
S43 S44 S46 S47 S50 S51 S52 S54 S55 S56 S58 S60 S61 S62 S63 S64
S65 S66 S67 S69 S70

35n70s1-52s:
S01 S03 S04 S07 S08 S09 S10 S11 S13 S15 S16 S18 S19 S20 S21 S22
S23 S25 S26 S27 **S28** S29 S30 S31 S32 S35 S36 S37 S39 S41 S42 S43
S44 S46 S47 S50 S51 S52 S54 S55 S56 S58 S60 S61 S62 S63 S64 S65
S66 S67 S69 S70

35n70s1-51s:
S01 S03 S04 S07 S08 **S09** S10 S11 S13 S15 S16 S18 S19 S20 S21 S22
S23 S25 S26 S27 S29 S30 S31 S32 S35 S36 S37 S39 S41 S42 S43 S44
S46 S47 S50 S51 S52 S54 S55 S56 S58 S60 S61 S62 S63 S64 S65 S66
S67 S69 S70

35n70s1-50s:
S01 S03 S04 S07 S08 S10 S11 S13 S15 S16 S18 S19 S20 S21 S22 S23
S25 S26 S27 S29 S30 S31 S32 S35 S36 S37 S39 S41 S42 S43 S44 S46
S47 S50 S51 S52 **S54** S55 S56 S58 S60 S61 S62 S63 S64 S65 S66 S67
S69 S70

35n70s1-49s:
S01 S03 S04 S07 S08 S10 S11 S13 S15 S16 S18 S19 S20 S21 S22 **S23**
S25 S26 S27 S29 S30 S31 S32 S35 S36 S37 S39 S41 S42 S43 S44 S46
S47 S50 S51 S52 S55 S56 S58 S60 S61 S62 S63 S64 S65 S66 S67 S69
S70

35n70s1-48s:
S01 S03 S04 S07 S08 S10 S11 S13 S15 S16 S18 S19 S20 **S21** S22 S25
S26 S27 S29 S30 S31 S32 S35 S36 S37 S39 S41 S42 S43 S44 S46 S47
S50 S51 S52 S55 S56 S58 S60 S61 S62 S63 S64 S65 S66 S67 S69 S70

35n70s1-47s:
S01 S03 S04 S07 S08 S10 S11 S13 S15 S16 S18 S19 S20 S22 S25 S26
S27 S29 S30 S31 S32 S35 S36 S37 S39 S41 **S42** S43 S44 S46 S47 S50
S51 S52 S55 S56 S58 S60 S61 S62 S63 S64 S65 S66 S67 S69 S70

35n70s1-46s:
S01 S03 S04 S07 S08 **S10** S11 S13 S15 S16 S18 S19 S20 S22 S25 S26
S27 S29 S30 S31 S32 S35 S36 S37 S39 S41 S43 S44 S46 S47 S50 S51
S52 S55 S56 S58 S60 S61 S62 S63 S64 S65 S66 S67 S69 S70

35n70s1-45s:
S01 S03 S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 S25 S26 S27
S29 S30 S31 S32 S35 S36 S37 S39 S41 S43 S44 S46 S47 S50 S51 S52
S55 S56 S58 S60 S61 S62 S63 S64 S65 S66 S67 S69 S70

35n70s1-44s:
S01 S03 S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 S25 S26 S27
S29 S30 S31 S32 S35 S36 S37 S39 S41 S43 S44 S46 S47 S50 S51 S52
S56 S58 S60 S61 S62 S63 S64 **S65** S66 S67 S69 S70

35n70s1-43s:
S01 S03 S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 **S25** S26 S27
S29 S30 S31 S32 S35 S36 S37 S39 S41 S43 S44 S46 S47 S50 S51 S52
S56 S58 S60 S61 S62 S63 S64 S66 S67 S69 S70

35n70s1-42s:
S01 **S03** S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 S26 S27 S29
S30 S31 S32 S35 S36 S37 S39 S41 S43 S44 S46 S47 S50 S51 S52 S56
S58 S60 S61 S62 S63 S64 S66 S67 S69 S70

35n70s1-41s:
S01 S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 **S26** S27 S29 S30
S31 S32 S35 S36 S37 S39 S41 S43 S44 S46 S47 S50 S51 S52 S56 S58
S60 S61 S62 S63 S64 S66 S67 S69 S70

35n70s1-40s:
S01 S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 S27 S29 S30 S31
S32 S35 S36 S37 S39 **S41** S43 S44 S46 S47 S50 S51 S52 S56 S58 S60
S61 S62 S63 S64 S66 S67 S69 S70

35n70s1-39s:
S01 S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 S27 S29 S30 S31
S32 S35 S36 S37 S39 S43 S44 S46 S47 S50 S51 S52 S56 S58 S60 S61
S62 S63 S64 **S66** S67 S69 S70

35n70s1-38s:

S01 S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 S27 S29 S30 S31
S32 S35 S36 S37 S39 S43 S44 S46 S47 S50 **S51** S52 S56 S58 S60 S61
S62 S63 S64 S67 S69 S70

35n70s1-37s:

S01 S04 S07 S08 S11 S13 S15 S16 S18 S19 S20 S22 S27 S29 S30 S31
S32 S35 S36 S37 S39 S43 S44 S46 S47 S50 S52 S56 S58 S60 S61 S62
S63 S64 S67 S69 S70

A.6 40N80S1 NETWORK FAMILY

```
NAME: 40n80s1
DATE LAST MODIFIED: Wednesday, July 25, 2001 2:56:43 PM MDT
MODIFIED BY: MeshBuilder(c)

NODE   X     Y     SIZE
N01   300    55     3
N02   363    90     4
N03   403    56     3
N04   437   104     3
N05   399   129     3
N06   307   126     3
N07   275   168     5
N08   363   160     4
N09   470   157     5
N10   389   255     4
N11   479   216     5
N12   505   266     3
N13   358   219     5
N14   298   206     3
N15   267   226     4
N16   360   314     8
N17   524   323     3
N18   557   379     3
N19   574   448     3
N20   419   337     4
N21   253   296     4
N22   206   184     3
N23   223   334     3
N24   190   425     4
N25   309   390     6
N26   424   385     5
N27   356   429     4
N28   283   444     5
N29   218   478     6
N30   388   486     4
N31   455   446     5
N32   531   497     3
N33   434   535     3
N34   324   498     5
N35   200   549     3
N36   285   563     4
N37   380   564     4
N38   485   599     4
N39   218   591     3
N40   344   610     4

SPAN   O     D     LENGTH
S01   N01   N02    72.069
S02   N01   N06    71.344
S03   N02   N03    52.498
S04   N02   N06    66.573
S05   N03   N04    58.822
S06   N04   N02    75.313
S07   N04   N09    62.434
S08   N05   N03    73.110
S09   N05   N07   129.988
S10   N05   N09    76.322
```

```
S11   N07   N06    52.802
S12   N07   N13    97.417
S13   N08   N01   122.450
S14   N08   N07    88.363
S15   N08   N09   107.042
S16   N08   N11   128.810
S17   N09   N10   127.142
S18   N10   N11    98.087
S19   N10   N16    65.742
S20   N11   N12    56.356
S21   N11   N20   135.059
S22   N12   N17    60.083
S23   N13   N09   128.016
S24   N13   N10    47.508
S25   N13   N16    95.021
S26   N13   N21   130.208
S27   N14   N07    44.418
S28   N15   N14    36.892
S29   N15   N16   128.035
S30   N15   N21    71.386
S31   N16   N11   154.159
S32   N16   N14   124.531
S33   N16   N20    63.325
S34   N16   N24   203.030
S35   N16   N34   187.489
S36   N17   N18    65.000
S37   N17   N38   278.742
S38   N18   N19    71.063
S39   N19   N31   119.017
S40   N19   N32    65.192
S41   N20   N12   111.521
S42   N21   N25   109.417
S43   N22   N15    74.061
S44   N22   N21   121.462
S45   N23   N22   150.960
S46   N24   N23    96.799
S47   N24   N25   124.040
S48   N25   N23   102.626
S49   N25   N26   115.109
S50   N25   N28    59.933
S51   N25   N29   126.590
S52   N26   N18   133.135
S53   N26   N20    48.260
S54   N26   N31    68.425
S55   N27   N26    80.994
S56   N27   N31   100.449
S57   N28   N27    74.525
S58   N28   N29    73.355
S59   N28   N34    67.801
S60   N29   N24    59.942
S61   N29   N34   107.870
S62   N29   N36   108.231
S63   N30   N27    65.368
S64   N31   N30    78.032
S65   N31   N33    91.444
S66   N32   N30   143.422
S67   N32   N38   111.893
S68   N33   N37    61.294
S69   N33   N38    81.835
S70   N34   N36    75.802
S71   N35   N28   133.843
```

```
S72   N35   N29    73.246
S73   N36   N40    75.432
S74   N37   N30    78.409
S75   N37   N34    86.556
S76   N38   N40   141.428
S77   N39   N35    45.695
S78   N39   N36    72.615
S79   N40   N37    58.412
S80   N40   N39   127.424
```

A.6.1 40N80S1 NETWORK FAMILY MEMBERS' SPAN LISTINGS

40n80s1:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S16
S17 S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32
S33 S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48
S49 S50 S51 S52 S53 S54 S55 S56 S57 S58 S59 S60 S61 S62 S63 S64
S65 S66 S67 S68 S69 S70 S71 S72 **S73** S74 S75 S76 S77 S78 S79 S80

40n80s1-79s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 **S16**
S17 S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32
S33 S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48
S49 S50 S51 S52 S53 S54 S55 S56 S57 S58 S59 S60 S61 S62 S63 S64
S65 S66 S67 S68 S69 S70 S71 S72 S74 S75 S76 S77 S78 S79 S80

40n80s1-78s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17
S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33
S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49
S50 S51 S52 S53 S54 S55 S56 S57 S58 S59 **S60** S61 S62 S63 S64 S65
S66 S67 S68 S69 S70 S71 S72 S74 S75 S76 S77 S78 S79 S80

40n80s1-77s:

S01 S02 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17
S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33
S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49
S50 S51 S52 S53 S54 **S55** S56 S57 S58 S59 S61 S62 S63 S64 S65 S66
S67 S68 S69 S70 S71 S72 S74 S75 S76 S77 S78 S79 S80

40n80s1-76s:

S01 **S02** S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17
S18 S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33
S34 S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49
S50 S51 S52 S53 S54 S56 S57 S58 S59 S61 S62 S63 S64 S65 S66 S67
S68 S69 S70 S71 S72 S74 S75 S76 S77 S78 S79 S80

40n80s1-75s:

S01 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18
S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50
S51 S52 S53 S54 S56 S57 S58 S59 S61 S62 S63 **S64** S65 S66 S67 S68
S69 S70 S71 S72 S74 S75 S76 S77 S78 S79 S80

40n80s1-74s:

S01 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18
S19 S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33 S34
S35 S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50
S51 S52 S53 S54 S56 S57 S58 S59 S61 S62 S63 S65 S66 S67 S68 S69
S70 S71 S72 S74 S75 S76 S77 S78 S79 S80

40n80s1-73s:
S01 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18
S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33 S34 S35
S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51
S52 S53 S54 S56 S57 S58 S59 S61 S62 **S63** S65 S66 S67 S68 S69 S70
S71 S72 S74 S75 S76 S77 S78 S79 S80

40n80s1-72s:
S01 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18
S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S32 S33 **S34** S35
S36 S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51
S52 S53 S54 S56 S57 S58 S59 S61 S62 S65 S66 S67 S68 S69 S70 S71
S72 S74 S75 S76 S77 S78 S79 S80

40n80s1-71s:
S01 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18
S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 **S32** S33 S35 S36
S37 S38 S39 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51 S52
S53 S54 S56 S57 S58 S59 S61 S62 S65 S66 S67 S68 S69 S70 S71 S72
S74 S75 S76 S77 S78 S79 S80

40n80s1-70s:
S01 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18
S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S35 S36 S37
S38 **S39** S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51 S52 S53
S54 S56 S57 S58 S59 S61 S62 S65 S66 S67 S68 S69 S70 S71 S72 S74
S75 S76 S77 S78 S79 S80

40n80s1-69s:
S01 S03 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18
S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S35 S36 S37
S38 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54
S56 S57 S58 S59 S61 S62 S65 S66 S67 S68 S69 S70 **S71** S72 S74 S75
S76 S77 S78 S79 S80

40n80s1-68s:
S01 **S03** S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18
S20 S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S35 S36 S37
S38 S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54
S56 S57 S58 S59 S61 S62 S65 S66 S67 S68 S69 S70 S72 S74 S75 S76
S77 S78 S79 S80

40n80s1-67s:
S01 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18 S20
S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 **S35** S36 S37 S38
S40 S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56
S57 S58 S59 S61 S62 S65 S66 S67 S68 S69 S70 S72 S74 S75 S76 S77
S78 S79 S80

40n80s1-66s:
S01 S04 S05 S06 S07 S08 S09 S10 S11 S12 S13 S14 S15 S17 S18 S20
S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40
S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57
S58 S59 S61 S62 S65 S66 S67 S68 S69 S70 S72 S74 S75 S76 S77 S78
S79 S80

40n80s1-65s:
S01 S04 S05 S06 S07 S08 **S09** S10 S11 S12 S13 S14 S15 S17 S18 S20
S21 S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40
S41 S42 S43 S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57
S58 S59 S61 S62 S65 S66 S67 S68 S69 S70 S72 S74 S75 S76 S77 S78
S80

40n80s1-64s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 S14 S15 S17 S18 S20 **S21**
S22 S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41
S42 S43 S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58
S59 S61 S62 S65 S66 S67 S68 S69 S70 S72 S74 S75 S76 S77 S78 S80

40n80s1-63s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 S14 S15 S17 S18 S20 S22
S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42
S43 S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59
S61 S62 S65 S66 S67 S68 S69 S70 S72 S74 S75 S76 S77 S78 S80

40n80s1-62s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 S14 S15 S17 S18 S20 S22
S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42
S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59 **S61**
S62 S65 S66 S67 S68 S69 S70 S72 S74 S75 S76 S77 S78 S80

40n80s1-61s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 S14 S15 S17 S18 S20 S22
S23 S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42
S44 S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59 S62
S65 S66 S67 S68 S69 S70 S72 S74 S75 S76 S77 S78 S80

40n80s1-60s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 S14 S15 S17 S18 S20 S22
S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42 S44
S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59 S62 S65
S66 **S67** S68 S69 S70 S72 S74 S75 S76 S77 S78 S80

40n80s1-59s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 S14 S15 S17 S18 S20 S22
S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42 S44
S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59 S62 S65
S66 S68 S69 S70 S72 S74 **S75** S76 S77 S78 S80

40n80s1-58s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 **S14** S15 S17 S18 S20 S22
S24 S25 S26 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42 S44
S45 S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59 S62 S65
S66 S68 S69 S70 S72 S74 S76 S77 S78 S80

40n80s1-57s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24
S25 **S26** S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42 S44 S45
S46 S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59 S62 S65 S66
S68 S69 S70 S72 S74 S76 S77 S78 S80

40n80s1-56s:
S01 S04 S05 S06 S07 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24
S25 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42 S44 S45 S46
S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59 S62 **S65** S66 S68
S69 S70 S72 S74 S76 S77 S78 S80

40n80s1-55s:
S01 S04 S05 S06 **S07** S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24
S25 S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42 S44 S45 S46
S47 S48 S49 S50 S51 S52 S53 S54 S56 S57 S58 S59 S62 S66 S68 S69
S70 S72 S74 S76 S77 S78 S80

40n80s1-54s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S29 S30 S31 S33 S36 S37 S38 S40 S41 S42 S44 S45 S46 S47
S48 S49 S50 **S51** S52 S53 S54 S56 S57 S58 S59 S62 S66 S68 S69 S70
S72 S74 S76 S77 S78 S80

40n80s1-53s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S29 S30 **S31** S33 S36 S37 S38 S40 S41 S42 S44 S45 S46 S47
S48 S49 S50 S52 S53 S54 S56 S57 S58 S59 S62 S66 S68 S69 S70 S72
S74 S76 S77 S78 S80

40n80s1-52s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S29 S30 S33 S36 S37 S38 S40 S41 S42 S44 S45 S46 S47 **S48**
S49 S50 S52 S53 S54 S56 S57 S58 S59 S62 S66 S68 S69 S70 S72 S74
S76 S77 S78 S80

40n80s1-51s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S29 S30 S33 S36 S37 S38 S40 S41 S42 S44 S45 S46 S47 S49
S50 S52 S53 S54 S56 S57 **S58** S59 S62 S66 S68 S69 S70 S72 S74 S76
S77 S78 S80

40n80s1-50s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 **S29** S30 S33 S36 S37 S38 S40 S41 S42 S44 S45 S46 S47 S49
S50 S52 S53 S54 S56 S57 S59 S62 S66 S68 S69 S70 S72 S74 S76 S77
S78 S80

40n80s1-49s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S30 S33 S36 **S37** S38 S40 S41 S42 S44 S45 S46 S47 S49 S50
S52 S53 S54 S56 S57 S59 S62 S66 S68 S69 S70 S72 S74 S76 S77 S78
S80

40n80s1-48s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S30 S33 S36 S38 S40 S41 **S42** S44 S45 S46 S47 S49 S50 S52
S53 S54 S56 S57 S59 S62 S66 S68 S69 S70 S72 S74 S76 S77 S78 S80

40n80s1-47s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S30 S33 S36 S38 S40 S41 S44 S45 S46 S47 **S49** S50 S52 S53
S54 S56 S57 S59 S62 S66 S68 S69 S70 S72 S74 S76 S77 S78 S80

40n80s1-46s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S30 S33 S36 S38 S40 S41 S44 S45 S46 S47 S50 S52 S53 S54
S56 S57 S59 S62 S66 S68 S69 S70 S72 S74 S76 S77 **S78** S80

40n80s1-45s:
S01 S04 S05 S06 S08 S10 S11 S12 S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S30 S33 S36 S38 S40 **S41** S44 S45 S46 S47 S50 S52 S53 S54
S56 S57 S59 S62 S66 S68 S69 S70 S72 S74 S76 S77 S80

40n80s1-44s:
S01 S04 S05 S06 S08 S10 S11 **S12** S13 S15 S17 S18 S20 S22 S24 S25
S27 S28 S30 S33 S36 S38 S40 S44 S45 S46 S47 S50 S52 S53 S54 S56
S57 S59 S62 S66 S68 S69 S70 S72 S74 S76 S77 S80

40n80s1-43s:
S01 S04 S05 S06 S08 S10 S11 S13 S15 S17 S18 S20 S22 S24 S25 S27
S28 S30 S33 S36 S38 S40 S44 S45 S46 S47 S50 **S52** S53 S54 S56 S57
S59 S62 S66 S68 S69 S70 S72 S74 S76 S77 S80

40n80s1-42s:

```
        S01 S04 S05 S06 S08 S10 S11 S13 S15 S17 S18 S20 S22 S24 S25 S27
        S28 S30 S33 S36 S38 S40 S44 S45 S46 S47 S50 S53 S54 S56 S57 S59
        S62 S66 S68 S69 S70 S72 S74 S76 S77 S80
```

APPENDIX B
NETWORK DEMAND FILES

B.1 15N30S1 NETWORK FAMILY

NAME: 15n30s1
DATE LAST MODIFIED: Wednesday, July 11, 2001 9:42:33 AM MDT
MODIFIED BY: MeshBuilder(c)

DEMAND	**O**	**D**	**UNITS**
D1	N01	N02	8
D2	N01	N03	4
D3	N01	N04	5
D4	N01	N05	6
D5	N01	N06	10
D6	N01	N07	3
D7	N01	N08	9
D8	N01	N09	9
D9	N01	N10	5
D10	N01	N11	10
D11	N01	N12	1
D12	N01	N13	1
D13	N01	N14	4
D14	N01	N15	4
D15	N02	N03	8
D16	N02	N04	2
D17	N02	N05	5
D18	N02	N06	1
D19	N02	N07	6
D20	N02	N08	1
D21	N02	N09	1
D22	N02	N10	3
D23	N02	N11	9
D24	N02	N12	6
D25	N02	N13	5
D26	N02	N14	5
D27	N02	N15	7
D28	N03	N04	2
D29	N03	N05	9
D30	N03	N06	2
D31	N03	N07	3
D32	N03	N08	6
D33	N03	N09	5
D34	N03	N10	9
D35	N03	N11	9
D36	N03	N12	6
D37	N03	N13	1
D38	N03	N14	4
D39	N03	N15	2
D40	N04	N05	3
D41	N04	N06	2
D42	N04	N07	4
D43	N04	N08	7
D44	N04	N09	9
D45	N04	N10	3
D46	N04	N11	5
D47	N04	N12	6
D48	N04	N13	4
D49	N04	N14	2
D50	N04	N15	1
D51	N05	N06	10
D52	N05	N07	7
D53	N05	N08	9
D54	N05	N09	6
D55	N05	N10	3
D56	N05	N11	3
D57	N05	N12	1
D58	N05	N13	1
D59	N05	N14	7
D60	N05	N15	3
D61	N06	N07	4
D62	N06	N08	6
D63	N06	N09	4
D64	N06	N10	5
D65	N06	N11	2
D66	N06	N12	5
D67	N06	N13	8
D68	N06	N14	9
D69	N06	N15	3
D70	N07	N08	5
D71	N07	N09	5
D72	N07	N10	7
D73	N07	N11	7
D74	N07	N12	1
D75	N07	N13	9
D76	N07	N14	4
D77	N07	N15	7
D78	N08	N09	2
D79	N08	N10	3
D80	N08	N11	6
D81	N08	N12	8
D82	N08	N13	8
D83	N08	N14	2
D84	N08	N15	10
D85	N09	N10	8
D86	N09	N11	6
D87	N09	N12	8
D88	N09	N13	10
D89	N09	N14	1
D90	N09	N15	1
D91	N10	N11	10
D92	N10	N12	2
D93	N10	N13	1
D94	N10	N14	7
D95	N10	N15	1
D96	N11	N12	5
D97	N11	N13	7
D98	N11	N14	1
D99	N11	N15	3
D100	N12	N13	3
D101	N12	N14	5
D102	N12	N15	1
D103	N13	N14	10
D104	N13	N15	3
D105	N14	N15	3

B.2 20N40S1 NETWORK FAMILY

NAME: 20n40s1
DATE LAST MODIFIED: Wednesday, July 11, 2001 9:42:03 AM MDT
MODIFIED BY: MeshBuilder(c)

DEMAND	O	D	UNITS
D1	N01	N02	9
D2	N01	N03	1
D3	N01	N04	7
D4	N01	N05	3
D5	N01	N06	3
D6	N01	N07	5
D7	N01	N08	9
D8	N01	N09	4
D9	N01	N10	7
D10	N01	N11	2
D11	N01	N12	6
D12	N01	N13	2
D13	N01	N14	3
D14	N01	N15	1
D15	N01	N16	2
D16	N01	N17	6
D17	N01	N18	1
D18	N01	N19	3
D19	N01	N20	10
D20	N02	N03	6
D21	N02	N04	7
D22	N02	N05	3
D23	N02	N06	6
D24	N02	N07	3
D25	N02	N08	2
D26	N02	N09	9
D27	N02	N10	1
D28	N02	N11	10
D29	N02	N12	2
D30	N02	N13	8
D31	N02	N14	6
D32	N02	N15	3
D33	N02	N16	10
D34	N02	N17	10
D35	N02	N18	4
D36	N02	N19	2
D37	N02	N20	3
D38	N03	N04	2
D39	N03	N05	4
D40	N03	N06	5
D41	N03	N07	8
D42	N03	N08	4
D43	N03	N09	9
D44	N03	N10	5
D45	N03	N11	8
D46	N03	N12	7
D47	N03	N13	2
D48	N03	N14	3
D49	N03	N15	8
D50	N03	N16	2
D51	N03	N17	1
D52	N03	N18	3
D53	N03	N19	5
D54	N03	N20	8
D55	N04	N05	5
D56	N04	N06	2
D57	N04	N07	5
D58	N04	N08	8
D59	N04	N09	6
D60	N04	N10	5
D61	N04	N11	3
D62	N04	N12	7
D63	N04	N13	10
D64	N04	N14	6
D65	N04	N15	2
D66	N04	N16	9
D67	N04	N17	5
D68	N04	N18	9
D69	N04	N19	9
D70	N04	N20	6
D71	N05	N06	9
D72	N05	N07	5
D73	N05	N08	7
D74	N05	N09	6
D75	N05	N10	8
D76	N05	N11	5
D77	N05	N12	5
D78	N05	N13	5
D79	N05	N14	8
D80	N05	N15	1
D81	N05	N16	7
D82	N05	N17	10
D83	N05	N18	9
D84	N05	N19	8
D85	N05	N20	8
D86	N06	N07	10
D87	N06	N08	5
D88	N06	N09	2
D89	N06	N10	3
D90	N06	N11	3
D91	N06	N12	4
D92	N06	N13	7
D93	N06	N14	10
D94	N06	N15	6
D95	N06	N16	1
D96	N06	N17	1
D97	N06	N18	10
D98	N06	N19	7
D99	N06	N20	2
D100	N07	N08	4
D101	N07	N09	4
D102	N07	N10	9
D103	N07	N11	3
D104	N07	N12	4
D105	N07	N13	8
D106	N07	N14	4
D107	N07	N15	1
D108	N07	N16	6
D109	N07	N17	9
D110	N07	N18	3
D111	N07	N19	10
D112	N07	N20	1
D113	N08	N09	4
D114	N08	N10	3
D115	N08	N11	6
D116	N08	N12	10
D117	N08	N13	5
D118	N08	N14	8
D119	N08	N15	8
D120	N08	N16	5
D121	N08	N17	9
D122	N08	N18	6
D123	N08	N19	4
D124	N08	N20	3
D125	N09	N10	8
D126	N09	N11	8
D127	N09	N12	8
D128	N09	N13	3
D129	N09	N14	2
D130	N09	N15	10
D131	N09	N16	10
D132	N09	N17	7
D133	N09	N18	5
D134	N09	N19	4
D135	N09	N20	10
D136	N10	N11	8
D137	N10	N12	5
D138	N10	N13	4
D139	N10	N14	10
D140	N10	N15	1
D141	N10	N16	8
D142	N10	N17	9
D143	N10	N18	5
D144	N10	N19	1
D145	N10	N20	1
D146	N11	N12	2
D147	N11	N13	2
D148	N11	N14	7
D149	N11	N15	8
D150	N11	N16	10
D151	N11	N17	5
D152	N11	N18	1
D153	N11	N19	10

DEMAND	O	D	UNITS
D154	N11	N20	2
D155	N12	N13	8
D156	N12	N14	6
D157	N12	N15	7
D158	N12	N16	9
D159	N12	N17	2
D160	N12	N18	4
D161	N12	N19	7
D162	N12	N20	9
D163	N13	N14	9
D164	N13	N15	2
D165	N13	N16	3
D166	N13	N17	8
D167	N13	N18	7
D168	N13	N19	5
D169	N13	N20	8
D170	N14	N15	3
D171	N14	N16	9
D172	N14	N17	3
D173	N14	N18	8
D174	N14	N19	6
D175	N14	N20	1
D176	N15	N16	4
D177	N15	N17	6
D178	N15	N18	10
D179	N15	N19	1
D180	N15	N20	6
D181	N16	N17	1
D182	N16	N18	9
D183	N16	N19	6
D184	N16	N20	5
D185	N17	N18	4
D186	N17	N19	7
D187	N17	N20	1
D188	N18	N19	2
D189	N18	N20	1
D190	N19	N20	9

B.3 25N50S1 NETWORK FAMILY

NAME: 25n50s1
DATE LAST MODIFIED: Wednesday, July 11, 2001 9:41:32 AM MDT
MODIFIED BY: MeshBuilder(c)

DEMAND	O	D	UNITS
D1	N01	N02	2
D2	N01	N03	1
D3	N01	N04	9
D4	N01	N05	7
D5	N01	N06	1
D6	N01	N07	2
D7	N01	N08	4
D8	N01	N09	10
D9	N01	N10	1
D10	N01	N11	3
D11	N01	N12	4
D12	N01	N13	3
D13	N01	N14	4
D14	N01	N15	3
D15	N01	N16	10
D16	N01	N17	1
D17	N01	N18	6
D18	N01	N19	1
D19	N01	N20	10
D20	N01	N21	3
D21	N01	N22	9
D22	N01	N23	10
D23	N01	N24	9
D24	N01	N25	4
D25	N02	N03	8
D26	N02	N04	1
D27	N02	N05	5
D28	N02	N06	8
D29	N02	N07	4
D30	N02	N08	2
D31	N02	N09	6
D32	N02	N10	3
D33	N02	N11	9
D34	N02	N12	6
D35	N02	N13	3
D36	N02	N14	10
D37	N02	N15	4
D38	N02	N16	6
D39	N02	N17	3
D40	N02	N18	4
D41	N02	N19	10
D42	N02	N20	6
D43	N02	N21	3
D44	N02	N22	7
D45	N02	N23	7
D46	N02	N24	1
D47	N02	N25	1
D48	N03	N04	6
D49	N03	N05	4
D50	N03	N06	3
D51	N03	N07	6
D52	N03	N08	3
D53	N03	N09	5
D54	N03	N10	9
D55	N03	N11	1
D56	N03	N12	3
D57	N03	N13	1
D58	N03	N14	2
D59	N03	N15	6
D60	N03	N16	6
D61	N03	N17	9
D62	N03	N18	3
D63	N03	N19	5
D64	N03	N20	1
D65	N03	N21	5
D66	N03	N22	10
D67	N03	N23	6
D68	N03	N24	8
D69	N03	N25	3
D70	N04	N05	1
D71	N04	N06	5
D72	N04	N07	7
D73	N04	N08	6
D74	N04	N09	5
D75	N04	N10	4
D76	N04	N11	5
D77	N04	N12	3
D78	N04	N13	1
D79	N04	N14	3
D80	N04	N15	3
D81	N04	N16	8
D82	N04	N17	2
D83	N04	N18	7
D84	N04	N19	9
D85	N04	N20	9
D86	N04	N21	8
D87	N04	N22	5
D88	N04	N23	3
D89	N04	N24	8
D90	N04	N25	1
D91	N05	N06	5
D92	N05	N07	4
D93	N05	N08	4
D94	N05	N09	10
D95	N05	N10	8
D96	N05	N11	8
D97	N05	N12	4
D98	N05	N13	2
D99	N05	N14	8
D100	N05	N15	9
D101	N05	N16	8
D102	N05	N17	8
D103	N05	N18	6
D104	N05	N19	7
D105	N05	N20	2
D106	N05	N21	4
D107	N05	N22	6
D108	N05	N23	1
D109	N05	N24	4
D110	N05	N25	6
D111	N06	N07	5
D112	N06	N08	4
D113	N06	N09	1
D114	N06	N10	4
D115	N06	N11	2
D116	N06	N12	6
D117	N06	N13	6
D118	N06	N14	9
D119	N06	N15	1
D120	N06	N16	9
D121	N06	N17	3
D122	N06	N18	3
D123	N06	N19	5
D124	N06	N20	5
D125	N06	N21	8
D126	N06	N22	3
D127	N06	N23	10
D128	N06	N24	8
D129	N06	N25	2
D130	N07	N08	9
D131	N07	N09	4
D132	N07	N10	1
D133	N07	N11	7
D134	N07	N12	2
D135	N07	N13	8
D136	N07	N14	3
D137	N07	N15	7
D138	N07	N16	6
D139	N07	N17	3
D140	N07	N18	9
D141	N07	N19	5
D142	N07	N20	1
D143	N07	N21	2
D144	N07	N22	5
D145	N07	N23	7
D146	N07	N24	4
D147	N07	N25	1
D148	N08	N09	8
D149	N08	N10	5
D150	N08	N11	3
D151	N08	N12	1
D152	N08	N13	5
D153	N08	N14	7

DEMAND	O	D	UNITS
D154	N08	N15	10
D155	N08	N16	7
D156	N08	N17	5
D157	N08	N18	6
D158	N08	N19	9
D159	N08	N20	5
D160	N08	N21	10
D161	N08	N22	7
D162	N08	N23	4
D163	N08	N24	6
D164	N08	N25	10
D165	N09	N10	2
D166	N09	N11	10
D167	N09	N12	5
D168	N09	N13	5
D169	N09	N14	10
D170	N09	N15	1
D171	N09	N16	4
D172	N09	N17	8
D173	N09	N18	2
D174	N09	N19	4
D175	N09	N20	5
D176	N09	N21	5
D177	N09	N22	3
D178	N09	N23	5
D179	N09	N24	9
D180	N09	N25	3
D181	N10	N11	7
D182	N10	N12	5
D183	N10	N13	4
D184	N10	N14	8
D185	N10	N15	2
D186	N10	N16	10
D187	N10	N17	2
D188	N10	N18	5
D189	N10	N19	3
D190	N10	N20	10
D191	N10	N21	6
D192	N10	N22	7
D193	N10	N23	9
D194	N10	N24	9
D195	N10	N25	9
D196	N11	N12	9
D197	N11	N13	2
D198	N11	N14	4
D199	N11	N15	7
D200	N11	N16	10
D201	N11	N17	4
D202	N11	N18	2
D203	N11	N19	8
D204	N11	N20	8
D205	N11	N21	7
D206	N11	N22	10
D207	N11	N23	2
D208	N11	N24	2
D209	N11	N25	5
D210	N12	N13	4
D211	N12	N14	4
D212	N12	N15	10
D213	N12	N16	2
D214	N12	N17	7
D215	N12	N18	3
D216	N12	N19	4
D217	N12	N20	3
D218	N12	N21	9
D219	N12	N22	8
D220	N12	N23	8
D221	N12	N24	6
D222	N12	N25	10
D223	N13	N14	1
D224	N13	N15	4
D225	N13	N16	4
D226	N13	N17	8
D227	N13	N18	9
D228	N13	N19	5
D229	N13	N20	9
D230	N13	N21	3
D231	N13	N22	6
D232	N13	N23	4
D233	N13	N24	8
D234	N13	N25	6
D235	N14	N15	5
D236	N14	N16	9
D237	N14	N17	3
D238	N14	N18	9
D239	N14	N19	8
D240	N14	N20	6
D241	N14	N21	5
D242	N14	N22	9
D243	N14	N23	8
D244	N14	N24	5
D245	N14	N25	9
D246	N15	N16	4
D247	N15	N17	1
D248	N15	N18	5
D249	N15	N19	4
D250	N15	N20	5
D251	N15	N21	9
D252	N15	N22	6
D253	N15	N23	1
D254	N15	N24	8
D255	N15	N25	5
D256	N16	N17	4
D257	N16	N18	10
D258	N16	N19	8
D259	N16	N20	1
D260	N16	N21	6
D261	N16	N22	5
D262	N16	N23	7
D263	N16	N24	1
D264	N16	N25	7
D265	N17	N18	1
D266	N17	N19	3
D267	N17	N20	3
D268	N17	N21	9
D269	N17	N22	10
D270	N17	N23	2
D271	N17	N24	10
D272	N17	N25	3
D273	N18	N19	3
D274	N18	N20	9
D275	N18	N21	1
D276	N18	N22	4
D277	N18	N23	5
D278	N18	N24	9
D279	N18	N25	10
D280	N19	N20	8
D281	N19	N21	4
D282	N19	N22	5
D283	N19	N23	9
D284	N19	N24	10
D285	N19	N25	4
D286	N20	N21	1
D287	N20	N22	1
D288	N20	N23	1
D289	N20	N24	6
D290	N20	N25	10
D291	N21	N22	3
D292	N21	N23	6
D293	N21	N24	10
D294	N21	N25	8
D295	N22	N23	2
D296	N22	N24	5
D297	N22	N25	8
D298	N23	N24	7
D299	N23	N25	1
D300	N24	N25	7

B.4 30N60S1 NETWORK FAMILY

NAME: 30n60s1
DATE LAST MODIFIED: Wednesday, July 11, 2001 9:40:55 AM MDT
MODIFIED BY: MeshBuilder(c)

DEMAND	O	D	UNITS
D1	N01	N02	5
D2	N01	N03	4
D3	N01	N04	4
D4	N01	N05	4
D5	N01	N06	8
D6	N01	N07	10
D7	N01	N08	10
D8	N01	N09	10
D9	N01	N10	6
D10	N01	N11	2
D11	N01	N12	3
D12	N01	N13	5
D13	N01	N14	6
D14	N01	N15	8
D15	N01	N16	8
D16	N01	N17	6
D17	N01	N18	7
D18	N01	N19	10
D19	N01	N20	2
D20	N01	N21	1
D21	N01	N22	6
D22	N01	N23	3
D23	N01	N24	10
D24	N01	N25	4
D25	N01	N26	2
D26	N01	N27	3
D27	N01	N28	10
D28	N01	N29	3
D29	N01	N30	8
D30	N02	N03	3
D31	N02	N04	6
D32	N02	N05	9
D33	N02	N06	1
D34	N02	N07	4
D35	N02	N08	10
D36	N02	N09	2
D37	N02	N10	1
D38	N02	N11	5
D39	N02	N12	1
D40	N02	N13	8
D41	N02	N14	2
D42	N02	N15	1
D43	N02	N16	5
D44	N02	N17	10
D45	N02	N18	5
D46	N02	N19	10
D47	N02	N20	4
D48	N02	N21	7
D49	N02	N22	7
D50	N02	N23	10
D51	N02	N24	4
D52	N02	N25	2
D53	N02	N26	4
D54	N02	N27	10
D55	N02	N28	8
D56	N02	N29	2
D57	N02	N30	9
D58	N03	N04	9
D59	N03	N05	3
D60	N03	N06	9
D61	N03	N07	3
D62	N03	N08	7
D63	N03	N09	2
D64	N03	N10	10
D65	N03	N11	1
D66	N03	N12	3
D67	N03	N13	7
D68	N03	N14	2
D69	N03	N15	6
D70	N03	N16	4
D71	N03	N17	2
D72	N03	N18	9
D73	N03	N19	3
D74	N03	N20	5
D75	N03	N21	3
D76	N03	N22	2
D77	N03	N23	5
D78	N03	N24	6
D79	N03	N25	8
D80	N03	N26	1
D81	N03	N27	5
D82	N03	N28	5
D83	N03	N29	3
D84	N03	N30	5
D85	N04	N05	1
D86	N04	N06	2
D87	N04	N07	4
D88	N04	N08	10
D89	N04	N09	1
D90	N04	N10	8
D91	N04	N11	1
D92	N04	N12	4
D93	N04	N13	1
D94	N04	N14	7
D95	N04	N15	8
D96	N04	N16	3
D97	N04	N17	1
D98	N04	N18	9
D99	N04	N19	10
D100	N04	N20	5
D101	N04	N21	4
D102	N04	N22	8
D103	N04	N23	3
D104	N04	N24	10
D105	N04	N25	10
D106	N04	N26	9
D107	N04	N27	9
D108	N04	N28	7
D109	N04	N29	8
D110	N04	N30	10
D111	N05	N06	4
D112	N05	N07	7
D113	N05	N08	8
D114	N05	N09	4
D115	N05	N10	7
D116	N05	N11	2
D117	N05	N12	8
D118	N05	N13	5
D119	N05	N14	8
D120	N05	N15	7
D121	N05	N16	6
D122	N05	N17	9
D123	N05	N18	5
D124	N05	N19	9
D125	N05	N20	8
D126	N05	N21	4
D127	N05	N22	8
D128	N05	N23	1
D129	N05	N24	7
D130	N05	N25	10
D131	N05	N26	9
D132	N05	N27	10
D133	N05	N28	2
D134	N05	N29	8
D135	N05	N30	3
D136	N06	N07	7
D137	N06	N08	4
D138	N06	N09	10
D139	N06	N10	1
D140	N06	N11	4
D141	N06	N12	1
D142	N06	N13	1
D143	N06	N14	8
D144	N06	N15	9
D145	N06	N16	10
D146	N06	N17	5
D147	N06	N18	5
D148	N06	N19	5
D149	N06	N20	1
D150	N06	N21	3
D151	N06	N22	1
D152	N06	N23	8
D153	N06	N24	4

DEMAND	O	D	UNITS
D154	N06	N25	3
D155	N06	N26	6
D156	N06	N27	6
D157	N06	N28	9
D158	N06	N29	5
D159	N06	N30	4
D160	N07	N08	10
D161	N07	N09	1
D162	N07	N10	5
D163	N07	N11	5
D164	N07	N12	1
D165	N07	N13	10
D166	N07	N14	7
D167	N07	N15	2
D168	N07	N16	5
D169	N07	N17	3
D170	N07	N18	10
D171	N07	N19	10
D172	N07	N20	8
D173	N07	N21	3
D174	N07	N22	9
D175	N07	N23	8
D176	N07	N24	3
D177	N07	N25	5
D178	N07	N26	6
D179	N07	N27	5
D180	N07	N28	5
D181	N07	N29	4
D182	N07	N30	8
D183	N08	N09	5
D184	N08	N10	7
D185	N08	N11	6
D186	N08	N12	8
D187	N08	N13	7
D188	N08	N14	5
D189	N08	N15	5
D190	N08	N16	8
D191	N08	N17	7
D192	N08	N18	5
D193	N08	N19	10
D194	N08	N20	9
D195	N08	N21	2
D196	N08	N22	3
D197	N08	N23	10
D198	N08	N24	1
D199	N08	N25	6
D200	N08	N26	4
D201	N08	N27	9
D202	N08	N28	3
D203	N08	N29	4
D204	N08	N30	10
D205	N09	N10	7
D206	N09	N11	8
D207	N09	N12	4
D208	N09	N13	10
D209	N09	N14	2
D210	N09	N15	10
D211	N09	N16	7
D212	N09	N17	9
D213	N09	N18	9
D214	N09	N19	5
D215	N09	N20	5
D216	N09	N21	5
D217	N09	N22	8
D218	N09	N23	9
D219	N09	N24	8
D220	N09	N25	4
D221	N09	N26	7
D222	N09	N27	9
D223	N09	N28	1
D224	N09	N29	4
D225	N09	N30	9
D226	N10	N11	1
D227	N10	N12	9
D228	N10	N13	4
D229	N10	N14	5
D230	N10	N15	4
D231	N10	N16	8
D232	N10	N17	6
D233	N10	N18	9
D234	N10	N19	2
D235	N10	N20	1
D236	N10	N21	2
D237	N10	N22	5
D238	N10	N23	1
D239	N10	N24	3
D240	N10	N25	5
D241	N10	N26	7
D242	N10	N27	7
D243	N10	N28	5
D244	N10	N29	9
D245	N10	N30	6
D246	N11	N12	8
D247	N11	N13	6
D248	N11	N14	5
D249	N11	N15	5
D250	N11	N16	2
D251	N11	N17	7
D252	N11	N18	7
D253	N11	N19	2
D254	N11	N20	5
D255	N11	N21	6
D256	N11	N22	1
D257	N11	N23	3
D258	N11	N24	2
D259	N11	N25	9
D260	N11	N26	2
D261	N11	N27	7
D262	N11	N28	5
D263	N11	N29	4
D264	N11	N30	6
D265	N12	N13	6
D266	N12	N14	5
D267	N12	N15	2
D268	N12	N16	8
D269	N12	N17	5
D270	N12	N18	3
D271	N12	N19	2
D272	N12	N20	7
D273	N12	N21	3
D274	N12	N22	1
D275	N12	N23	8
D276	N12	N24	10
D277	N12	N25	3
D278	N12	N26	9
D279	N12	N27	4
D280	N12	N28	8
D281	N12	N29	5
D282	N12	N30	7
D283	N13	N14	8
D284	N13	N15	10
D285	N13	N16	9
D286	N13	N17	1
D287	N13	N18	5
D288	N13	N19	3
D289	N13	N20	2
D290	N13	N21	5
D291	N13	N22	9
D292	N13	N23	5
D293	N13	N24	4
D294	N13	N25	7
D295	N13	N26	3
D296	N13	N27	2
D297	N13	N28	8
D298	N13	N29	4
D299	N13	N30	4
D300	N14	N15	8
D301	N14	N16	4
D302	N14	N17	2
D303	N14	N18	3
D304	N14	N19	4
D305	N14	N20	6
D306	N14	N21	7
D307	N14	N22	1
D308	N14	N23	9
D309	N14	N24	1
D310	N14	N25	6
D311	N14	N26	3
D312	N14	N27	2
D313	N14	N28	10
D314	N14	N29	2
D315	N14	N30	2
D316	N15	N16	9
D317	N15	N17	2
D318	N15	N18	10
D319	N15	N19	7
D320	N15	N20	8
D321	N15	N21	7
D322	N15	N22	3
D323	N15	N23	10
D324	N15	N24	5
D325	N15	N25	4
D326	N15	N26	10
D327	N15	N27	5
D328	N15	N28	3
D329	N15	N29	2
D330	N15	N30	4
D331	N16	N17	7
D332	N16	N18	10
D333	N16	N19	2

DEMAND	**O**	**D**	**UNITS**
D334	N16	N20	1
D335	N16	N21	9
D336	N16	N22	4
D337	N16	N23	8
D338	N16	N24	1
D339	N16	N25	9
D340	N16	N26	1
D341	N16	N27	3
D342	N16	N28	10
D343	N16	N29	10
D344	N16	N30	2
D345	N17	N18	5
D346	N17	N19	10
D347	N17	N20	9
D348	N17	N21	6
D349	N17	N22	2
D350	N17	N23	9
D351	N17	N24	7
D352	N17	N25	1
D353	N17	N26	1
D354	N17	N27	7
D355	N17	N28	10
D356	N17	N29	7
D357	N17	N30	4
D358	N18	N19	7
D359	N18	N20	5
D360	N18	N21	3
D361	N18	N22	7
D362	N18	N23	5
D363	N18	N24	9
D364	N18	N25	3
D365	N18	N26	2
D366	N18	N27	6
D367	N18	N28	10
D368	N18	N29	8
D369	N18	N30	7
D370	N19	N20	3
D371	N19	N21	7
D372	N19	N22	2
D373	N19	N23	3
D374	N19	N24	8
D375	N19	N25	4
D376	N19	N26	6
D377	N19	N27	9
D378	N19	N28	5
D379	N19	N29	4
D380	N19	N30	10
D381	N20	N21	1
D382	N20	N22	5
D383	N20	N23	4
D384	N20	N24	1
D385	N20	N25	4
D386	N20	N26	8
D387	N20	N27	10
D388	N20	N28	3
D389	N20	N29	9
D390	N20	N30	9
D391	N21	N22	2
D392	N21	N23	3
D393	N21	N24	9
D394	N21	N25	9
D395	N21	N26	5
D396	N21	N27	6
D397	N21	N28	7
D398	N21	N29	10
D399	N21	N30	8
D400	N22	N23	7
D401	N22	N24	3
D402	N22	N25	5
D403	N22	N26	8
D404	N22	N27	10
D405	N22	N28	6
D406	N22	N29	5
D407	N22	N30	10
D408	N23	N24	5
D409	N23	N25	7
D410	N23	N26	7
D411	N23	N27	8
D412	N23	N28	6
D413	N23	N29	9
D414	N23	N30	3
D415	N24	N25	3
D416	N24	N26	6
D417	N24	N27	1
D418	N24	N28	8
D419	N24	N29	3
D420	N24	N30	7
D421	N25	N26	1
D422	N25	N27	4
D423	N25	N28	10
D424	N25	N29	4
D425	N25	N30	7
D426	N26	N27	5
D427	N26	N28	10
D428	N26	N29	1
D429	N26	N30	6
D430	N27	N28	10
D431	N27	N29	10
D432	N27	N30	7
D433	N28	N29	5
D434	N28	N30	7
D435	N29	N30	7

B.5 35N70S1 NETWORK FAMILY

```
NAME: 35n70s1
DATE LAST MODIFIED: Wednesday, July 11, 2001 9:40:20 AM MDT
MODIFIED BY: MeshBuilder(c)
```

DEMAND	O	D	UNITS
D1	N01	N02	5
D2	N01	N03	6
D3	N01	N04	2
D4	N01	N05	4
D5	N01	N06	3
D6	N01	N07	3
D7	N01	N08	4
D8	N01	N09	3
D9	N01	N10	4
D10	N01	N11	7
D11	N01	N12	6
D12	N01	N13	5
D13	N01	N14	3
D14	N01	N15	7
D15	N01	N16	4
D16	N01	N17	10
D17	N01	N18	6
D18	N01	N19	5
D19	N01	N20	6
D20	N01	N21	4
D21	N01	N22	9
D22	N01	N23	1
D23	N01	N24	2
D24	N01	N25	5
D25	N01	N26	2
D26	N01	N27	10
D27	N01	N28	6
D28	N01	N29	8
D29	N01	N30	7
D30	N01	N31	9
D31	N01	N32	9
D32	N01	N33	5
D33	N01	N34	6
D34	N01	N35	3
D35	N02	N03	8
D36	N02	N04	7
D37	N02	N05	8
D38	N02	N06	1
D39	N02	N07	10
D40	N02	N08	7
D41	N02	N09	4
D42	N02	N10	9
D43	N02	N11	10
D44	N02	N12	7
D45	N02	N13	10
D46	N02	N14	9
D47	N02	N15	7
D48	N02	N16	7
D49	N02	N17	9
D50	N02	N18	4
D51	N02	N19	3
D52	N02	N20	10
D53	N02	N21	6
D54	N02	N22	2
D55	N02	N23	4
D56	N02	N24	2
D57	N02	N25	8
D58	N02	N26	2
D59	N02	N27	2
D60	N02	N28	2
D61	N02	N29	9
D62	N02	N30	8
D63	N02	N31	4
D64	N02	N32	10
D65	N02	N33	10
D66	N02	N34	7
D67	N02	N35	2
D68	N03	N04	3
D69	N03	N05	3
D70	N03	N06	2
D71	N03	N07	2
D72	N03	N08	10
D73	N03	N09	9
D74	N03	N10	1
D75	N03	N11	10
D76	N03	N12	2
D77	N03	N13	2
D78	N03	N14	1
D79	N03	N15	4
D80	N03	N16	4
D81	N03	N17	9
D82	N03	N18	3
D83	N03	N19	5
D84	N03	N20	9
D85	N03	N21	10
D86	N03	N22	2
D87	N03	N23	7
D88	N03	N24	3
D89	N03	N25	9
D90	N03	N26	4
D91	N03	N27	3
D92	N03	N28	2
D93	N03	N29	7
D94	N03	N30	5
D95	N03	N31	7
D96	N03	N32	2
D97	N03	N33	3
D98	N03	N34	3
D99	N03	N35	3
D100	N04	N05	4
D101	N04	N06	10
D102	N04	N07	6
D103	N04	N08	2
D104	N04	N09	10
D105	N04	N10	6
D106	N04	N11	6
D107	N04	N12	4
D108	N04	N13	9
D109	N04	N14	2
D110	N04	N15	5
D111	N04	N16	3
D112	N04	N17	7
D113	N04	N18	4
D114	N04	N19	7
D115	N04	N20	8
D116	N04	N21	10
D117	N04	N22	9
D118	N04	N23	2
D119	N04	N24	1
D120	N04	N25	6
D121	N04	N26	1
D122	N04	N27	1
D123	N04	N28	1
D124	N04	N29	9
D125	N04	N30	6
D126	N04	N31	7
D127	N04	N32	7
D128	N04	N33	4
D129	N04	N34	4
D130	N04	N35	5
D131	N05	N06	3
D132	N05	N07	1
D133	N05	N08	3
D134	N05	N09	10
D135	N05	N10	10
D136	N05	N11	8
D137	N05	N12	10
D138	N05	N13	1
D139	N05	N14	1
D140	N05	N15	5
D141	N05	N16	10
D142	N05	N17	3
D143	N05	N18	8
D144	N05	N19	6
D145	N05	N20	6
D146	N05	N21	8
D147	N05	N22	9
D148	N05	N23	4
D149	N05	N24	2
D150	N05	N25	3
D151	N05	N26	9
D152	N05	N27	3
D153	N05	N28	1

DEMAND	O	D	UNITS
D154	N05	N29	9
D155	N05	N30	1
D156	N05	N31	5
D157	N05	N32	2
D158	N05	N33	6
D159	N05	N34	3
D160	N05	N35	3
D161	N06	N07	9
D162	N06	N08	2
D163	N06	N09	4
D164	N06	N10	10
D165	N06	N11	2
D166	N06	N12	9
D167	N06	N13	4
D168	N06	N14	1
D169	N06	N15	2
D170	N06	N16	8
D171	N06	N17	4
D172	N06	N18	1
D173	N06	N19	6
D174	N06	N20	3
D175	N06	N21	4
D176	N06	N22	4
D177	N06	N23	8
D178	N06	N24	8
D179	N06	N25	9
D180	N06	N26	10
D181	N06	N27	8
D182	N06	N28	2
D183	N06	N29	3
D184	N06	N30	10
D185	N06	N31	1
D186	N06	N32	3
D187	N06	N33	2
D188	N06	N34	8
D189	N06	N35	7
D190	N07	N08	4
D191	N07	N09	3
D192	N07	N10	1
D193	N07	N11	7
D194	N07	N12	9
D195	N07	N13	9
D196	N07	N14	3
D197	N07	N15	8
D198	N07	N16	8
D199	N07	N17	5
D200	N07	N18	2
D201	N07	N19	5
D202	N07	N20	7
D203	N07	N21	7
D204	N07	N22	2
D205	N07	N23	9
D206	N07	N24	10
D207	N07	N25	3
D208	N07	N26	4
D209	N07	N27	8
D210	N07	N28	9
D211	N07	N29	8
D212	N07	N30	6
D213	N07	N31	7

DEMAND	O	D	UNITS
D214	N07	N32	8
D215	N07	N33	3
D216	N07	N34	4
D217	N07	N35	7
D218	N08	N09	8
D219	N08	N10	4
D220	N08	N11	2
D221	N08	N12	5
D222	N08	N13	10
D223	N08	N14	7
D224	N08	N15	2
D225	N08	N16	10
D226	N08	N17	2
D227	N08	N18	4
D228	N08	N19	8
D229	N08	N20	9
D230	N08	N21	6
D231	N08	N22	6
D232	N08	N23	7
D233	N08	N24	6
D234	N08	N25	1
D235	N08	N26	10
D236	N08	N27	1
D237	N08	N28	8
D238	N08	N29	2
D239	N08	N30	6
D240	N08	N31	1
D241	N08	N32	6
D242	N08	N33	8
D243	N08	N34	1
D244	N08	N35	2
D245	N09	N10	3
D246	N09	N11	1
D247	N09	N12	7
D248	N09	N13	4
D249	N09	N14	3
D250	N09	N15	6
D251	N09	N16	4
D252	N09	N17	2
D253	N09	N18	3
D254	N09	N19	4
D255	N09	N20	5
D256	N09	N21	6
D257	N09	N22	4
D258	N09	N23	5
D259	N09	N24	9
D260	N09	N25	10
D261	N09	N26	10
D262	N09	N27	6
D263	N09	N28	5
D264	N09	N29	8
D265	N09	N30	4
D266	N09	N31	9
D267	N09	N32	1
D268	N09	N33	7
D269	N09	N34	10
D270	N09	N35	9
D271	N10	N11	7
D272	N10	N12	3
D273	N10	N13	1

DEMAND	O	D	UNITS
D274	N10	N14	6
D275	N10	N15	6
D276	N10	N16	4
D277	N10	N17	2
D278	N10	N18	9
D279	N10	N19	8
D280	N10	N20	7
D281	N10	N21	1
D282	N10	N22	10
D283	N10	N23	4
D284	N10	N24	1
D285	N10	N25	7
D286	N10	N26	1
D287	N10	N27	9
D288	N10	N28	7
D289	N10	N29	6
D290	N10	N30	5
D291	N10	N31	10
D292	N10	N32	10
D293	N10	N33	4
D294	N10	N34	6
D295	N10	N35	1
D296	N11	N12	9
D297	N11	N13	9
D298	N11	N14	1
D299	N11	N15	7
D300	N11	N16	10
D301	N11	N17	7
D302	N11	N18	7
D303	N11	N19	8
D304	N11	N20	5
D305	N11	N21	10
D306	N11	N22	1
D307	N11	N23	7
D308	N11	N24	10
D309	N11	N25	6
D310	N11	N26	10
D311	N11	N27	4
D312	N11	N28	8
D313	N11	N29	8
D314	N11	N30	10
D315	N11	N31	7
D316	N11	N32	10
D317	N11	N33	6
D318	N11	N34	9
D319	N11	N35	7
D320	N12	N13	9
D321	N12	N14	6
D322	N12	N15	6
D323	N12	N16	4
D324	N12	N17	4
D325	N12	N18	1
D326	N12	N19	3
D327	N12	N20	1
D328	N12	N21	10
D329	N12	N22	1
D330	N12	N23	6
D331	N12	N24	3
D332	N12	N25	2
D333	N12	N26	5

DEMAND	O	D	UNITS
D334	N12	N27	8
D335	N12	N28	2
D336	N12	N29	10
D337	N12	N30	5
D338	N12	N31	7
D339	N12	N32	8
D340	N12	N33	2
D341	N12	N34	10
D342	N12	N35	10
D343	N13	N14	2
D344	N13	N15	4
D345	N13	N16	7
D346	N13	N17	4
D347	N13	N18	2
D348	N13	N19	7
D349	N13	N20	5
D350	N13	N21	7
D351	N13	N22	8
D352	N13	N23	5
D353	N13	N24	2
D354	N13	N25	7
D355	N13	N26	5
D356	N13	N27	2
D357	N13	N28	10
D358	N13	N29	6
D359	N13	N30	8
D360	N13	N31	10
D361	N13	N32	3
D362	N13	N33	4
D363	N13	N34	4
D364	N13	N35	6
D365	N14	N15	8
D366	N14	N16	7
D367	N14	N17	4
D368	N14	N18	8
D369	N14	N19	5
D370	N14	N20	10
D371	N14	N21	9
D372	N14	N22	8
D373	N14	N23	10
D374	N14	N24	3
D375	N14	N25	6
D376	N14	N26	2
D377	N14	N27	7
D378	N14	N28	10
D379	N14	N29	2
D380	N14	N30	7
D381	N14	N31	10
D382	N14	N32	4
D383	N14	N33	9
D384	N14	N34	6
D385	N14	N35	8
D386	N15	N16	6
D387	N15	N17	6
D388	N15	N18	8
D389	N15	N19	2
D390	N15	N20	5
D391	N15	N21	5
D392	N15	N22	9
D393	N15	N23	2
D394	N15	N24	10
D395	N15	N25	10
D396	N15	N26	10
D397	N15	N27	2
D398	N15	N28	5
D399	N15	N29	3
D400	N15	N30	5
D401	N15	N31	7
D402	N15	N32	7
D403	N15	N33	6
D404	N15	N34	10
D405	N15	N35	3
D406	N16	N17	10
D407	N16	N18	4
D408	N16	N19	4
D409	N16	N20	6
D410	N16	N21	9
D411	N16	N22	8
D412	N16	N23	5
D413	N16	N24	3
D414	N16	N25	5
D415	N16	N26	10
D416	N16	N27	10
D417	N16	N28	10
D418	N16	N29	2
D419	N16	N30	6
D420	N16	N31	8
D421	N16	N32	9
D422	N16	N33	4
D423	N16	N34	1
D424	N16	N35	6
D425	N17	N18	8
D426	N17	N19	1
D427	N17	N20	6
D428	N17	N21	9
D429	N17	N22	7
D430	N17	N23	4
D431	N17	N24	5
D432	N17	N25	5
D433	N17	N26	3
D434	N17	N27	5
D435	N17	N28	10
D436	N17	N29	7
D437	N17	N30	1
D438	N17	N31	10
D439	N17	N32	9
D440	N17	N33	2
D441	N17	N34	9
D442	N17	N35	7
D443	N18	N19	1
D444	N18	N20	4
D445	N18	N21	1
D446	N18	N22	8
D447	N18	N23	3
D448	N18	N24	3
D449	N18	N25	2
D450	N18	N26	10
D451	N18	N27	2
D452	N18	N28	1
D453	N18	N29	4
D454	N18	N30	2
D455	N18	N31	3
D456	N18	N32	6
D457	N18	N33	3
D458	N18	N34	10
D459	N18	N35	8
D460	N19	N20	6
D461	N19	N21	2
D462	N19	N22	6
D463	N19	N23	6
D464	N19	N24	2
D465	N19	N25	8
D466	N19	N26	2
D467	N19	N27	6
D468	N19	N28	5
D469	N19	N29	7
D470	N19	N30	5
D471	N19	N31	4
D472	N19	N32	4
D473	N19	N33	3
D474	N19	N34	10
D475	N19	N35	10
D476	N20	N21	3
D477	N20	N22	10
D478	N20	N23	4
D479	N20	N24	8
D480	N20	N25	2
D481	N20	N26	5
D482	N20	N27	3
D483	N20	N28	8
D484	N20	N29	6
D485	N20	N30	3
D486	N20	N31	9
D487	N20	N32	6
D488	N20	N33	5
D489	N20	N34	3
D490	N20	N35	1
D491	N21	N22	1
D492	N21	N23	7
D493	N21	N24	10
D494	N21	N25	7
D495	N21	N26	7
D496	N21	N27	6
D497	N21	N28	1
D498	N21	N29	3
D499	N21	N30	6
D500	N21	N31	5
D501	N21	N32	6
D502	N21	N33	10
D503	N21	N34	6
D504	N21	N35	10
D505	N22	N23	5
D506	N22	N24	6
D507	N22	N25	7
D508	N22	N26	8
D509	N22	N27	7
D510	N22	N28	1
D511	N22	N29	8
D512	N22	N30	5
D513	N22	N31	3

DEMAND	O	D	UNITS
D514	N22	N32	1
D515	N22	N33	2
D516	N22	N34	7
D517	N22	N35	7
D518	N23	N24	5
D519	N23	N25	3
D520	N23	N26	10
D521	N23	N27	8
D522	N23	N28	3
D523	N23	N29	9
D524	N23	N30	2
D525	N23	N31	3
D526	N23	N32	6
D527	N23	N33	3
D528	N23	N34	6
D529	N23	N35	7
D530	N24	N25	3
D531	N24	N26	7
D532	N24	N27	10
D533	N24	N28	9
D534	N24	N29	3
D535	N24	N30	6
D536	N24	N31	1
D537	N24	N32	8
D538	N24	N33	6
D539	N24	N34	6
D540	N24	N35	4
D541	N25	N26	7
D542	N25	N27	7
D543	N25	N28	1
D544	N25	N29	5
D545	N25	N30	10
D546	N25	N31	4
D547	N25	N32	10
D548	N25	N33	10
D549	N25	N34	3
D550	N25	N35	9
D551	N26	N27	9
D552	N26	N28	5
D553	N26	N29	9
D554	N26	N30	3
D555	N26	N31	6
D556	N26	N32	7
D557	N26	N33	3
D558	N26	N34	6
D559	N26	N35	1
D560	N27	N28	9
D561	N27	N29	3
D562	N27	N30	2
D563	N27	N31	7
D564	N27	N32	4
D565	N27	N33	6
D566	N27	N34	2
D567	N27	N35	10
D568	N28	N29	8
D569	N28	N30	3
D570	N28	N31	3
D571	N28	N32	7
D572	N28	N33	6
D573	N28	N34	1
D574	N28	N35	9
D575	N29	N30	6
D576	N29	N31	5
D577	N29	N32	5
D578	N29	N33	10
D579	N29	N34	9
D580	N29	N35	2
D581	N30	N31	3
D582	N30	N32	6
D583	N30	N33	5
D584	N30	N34	2
D585	N30	N35	2
D586	N31	N32	6
D587	N31	N33	5
D588	N31	N34	7
D589	N31	N35	6
D590	N32	N33	9
D591	N32	N34	8
D592	N32	N35	3
D593	N33	N34	7
D594	N33	N35	7
D595	N34	N35	6

B.6 40N80S1 NETWORK FAMILY

NAME: 40n80s1
DATE LAST MODIFIED: Wednesday, July 11, 2001 9:38:36 AM MDT
MODIFIED BY: MeshBuilder(c)

DEMAND	O	D	UNITS
D1	N01	N02	2
D2	N01	N03	1
D3	N01	N04	6
D4	N01	N05	10
D5	N01	N06	7
D6	N01	N07	3
D7	N01	N08	7
D8	N01	N09	2
D9	N01	N10	10
D10	N01	N11	7
D11	N01	N12	9
D12	N01	N13	5
D13	N01	N14	4
D14	N01	N15	4
D15	N01	N16	2
D16	N01	N17	2
D17	N01	N18	9
D18	N01	N19	5
D19	N01	N20	4
D20	N01	N21	10
D21	N01	N22	4
D22	N01	N23	5
D23	N01	N24	5
D24	N01	N25	2
D25	N01	N26	6
D26	N01	N27	8
D27	N01	N28	8
D28	N01	N29	7
D29	N01	N30	2
D30	N01	N31	10
D31	N01	N32	5
D32	N01	N33	10
D33	N01	N34	1
D34	N01	N35	5
D35	N01	N36	5
D36	N01	N37	3
D37	N01	N38	7
D38	N01	N39	10
D39	N01	N40	7
D40	N02	N03	9
D41	N02	N04	7
D42	N02	N05	1
D43	N02	N06	6
D44	N02	N07	1
D45	N02	N08	4
D46	N02	N09	5
D47	N02	N10	7
D48	N02	N11	6
D49	N02	N12	4
D50	N02	N13	7
D51	N02	N14	2
D52	N02	N15	3
D53	N02	N16	8
D54	N02	N17	5
D55	N02	N18	8
D56	N02	N19	4
D57	N02	N20	2
D58	N02	N21	3
D59	N02	N22	8
D60	N02	N23	2
D61	N02	N24	2
D62	N02	N25	8
D63	N02	N26	6
D64	N02	N27	7
D65	N02	N28	2
D66	N02	N29	4
D67	N02	N30	1
D68	N02	N31	8
D69	N02	N32	7
D70	N02	N33	4
D71	N02	N34	7
D72	N02	N35	4
D73	N02	N36	1
D74	N02	N37	3
D75	N02	N38	9
D76	N02	N39	9
D77	N02	N40	1
D78	N03	N04	3
D79	N03	N05	10
D80	N03	N06	6
D81	N03	N07	3
D82	N03	N08	5
D83	N03	N09	3
D84	N03	N10	2
D85	N03	N11	2
D86	N03	N12	3
D87	N03	N13	10
D88	N03	N14	9
D89	N03	N15	5
D90	N03	N16	9
D91	N03	N17	10
D92	N03	N18	1
D93	N03	N19	6
D94	N03	N20	2
D95	N03	N21	8
D96	N03	N22	7
D97	N03	N23	1
D98	N03	N24	2
D99	N03	N25	3
D100	N03	N26	6
D101	N03	N27	5
D102	N03	N28	1
D103	N03	N29	7
D104	N03	N30	9
D105	N03	N31	9
D106	N03	N32	1
D107	N03	N33	4
D108	N03	N34	3
D109	N03	N35	4
D110	N03	N36	1
D111	N03	N37	10
D112	N03	N38	4
D113	N03	N39	1
D114	N03	N40	9
D115	N04	N05	2
D116	N04	N06	7
D117	N04	N07	1
D118	N04	N08	2
D119	N04	N09	5
D120	N04	N10	4
D121	N04	N11	1
D122	N04	N12	9
D123	N04	N13	4
D124	N04	N14	3
D125	N04	N15	9
D126	N04	N16	7
D127	N04	N17	7
D128	N04	N18	6
D129	N04	N19	9
D130	N04	N20	10
D131	N04	N21	6
D132	N04	N22	2
D133	N04	N23	7
D134	N04	N24	1
D135	N04	N25	3
D136	N04	N26	9
D137	N04	N27	7
D138	N04	N28	6
D139	N04	N29	10
D140	N04	N30	8
D141	N04	N31	8
D142	N04	N32	3
D143	N04	N33	9
D144	N04	N34	9
D145	N04	N35	9
D146	N04	N36	6
D147	N04	N37	2
D148	N04	N38	4
D149	N04	N39	7
D150	N04	N40	7
D151	N05	N06	7
D152	N05	N07	3
D153	N05	N08	10

DEMAND	**O**	**D**	**UNITS**
D154	N05	N09	4
D155	N05	N10	1
D156	N05	N11	9
D157	N05	N12	8
D158	N05	N13	9
D159	N05	N14	3
D160	N05	N15	8
D161	N05	N16	3
D162	N05	N17	1
D163	N05	N18	4
D164	N05	N19	5
D165	N05	N20	8
D166	N05	N21	1
D167	N05	N22	5
D168	N05	N23	9
D169	N05	N24	5
D170	N05	N25	2
D171	N05	N26	3
D172	N05	N27	1
D173	N05	N28	10
D174	N05	N29	8
D175	N05	N30	2
D176	N05	N31	5
D177	N05	N32	4
D178	N05	N33	5
D179	N05	N34	9
D180	N05	N35	10
D181	N05	N36	7
D182	N05	N37	6
D183	N05	N38	2
D184	N05	N39	7
D185	N05	N40	6
D186	N06	N07	10
D187	N06	N08	7
D188	N06	N09	1
D189	N06	N10	2
D190	N06	N11	9
D191	N06	N12	1
D192	N06	N13	6
D193	N06	N14	9
D194	N06	N15	9
D195	N06	N16	4
D196	N06	N17	9
D197	N06	N18	6
D198	N06	N19	1
D199	N06	N20	5
D200	N06	N21	6
D201	N06	N22	4
D202	N06	N23	1
D203	N06	N24	9
D204	N06	N25	9
D205	N06	N26	5
D206	N06	N27	5
D207	N06	N28	8
D208	N06	N29	3
D209	N06	N30	4
D210	N06	N31	8
D211	N06	N32	1
D212	N06	N33	4
D213	N06	N34	3
D214	N06	N35	8
D215	N06	N36	8
D216	N06	N37	5
D217	N06	N38	10
D218	N06	N39	5
D219	N06	N40	3
D220	N07	N08	7
D221	N07	N09	6
D222	N07	N10	8
D223	N07	N11	6
D224	N07	N12	1
D225	N07	N13	9
D226	N07	N14	8
D227	N07	N15	7
D228	N07	N16	8
D229	N07	N17	5
D230	N07	N18	2
D231	N07	N19	6
D232	N07	N20	2
D233	N07	N21	4
D234	N07	N22	3
D235	N07	N23	5
D236	N07	N24	7
D237	N07	N25	8
D238	N07	N26	10
D239	N07	N27	2
D240	N07	N28	3
D241	N07	N29	7
D242	N07	N30	1
D243	N07	N31	4
D244	N07	N32	6
D245	N07	N33	9
D246	N07	N34	6
D247	N07	N35	1
D248	N07	N36	6
D249	N07	N37	7
D250	N07	N38	6
D251	N07	N39	3
D252	N07	N40	9
D253	N08	N09	5
D254	N08	N10	9
D255	N08	N11	1
D256	N08	N12	3
D257	N08	N13	3
D258	N08	N14	3
D259	N08	N15	2
D260	N08	N16	9
D261	N08	N17	4
D262	N08	N18	7
D263	N08	N19	8
D264	N08	N20	9
D265	N08	N21	6
D266	N08	N22	4
D267	N08	N23	9
D268	N08	N24	3
D269	N08	N25	10
D270	N08	N26	1
D271	N08	N27	1
D272	N08	N28	4
D273	N08	N29	6
D274	N08	N30	8
D275	N08	N31	2
D276	N08	N32	5
D277	N08	N33	1
D278	N08	N34	10
D279	N08	N35	8
D280	N08	N36	1
D281	N08	N37	1
D282	N08	N38	10
D283	N08	N39	1
D284	N08	N40	5
D285	N09	N10	5
D286	N09	N11	9
D287	N09	N12	10
D288	N09	N13	2
D289	N09	N14	8
D290	N09	N15	9
D291	N09	N16	4
D292	N09	N17	5
D293	N09	N18	3
D294	N09	N19	10
D295	N09	N20	6
D296	N09	N21	9
D297	N09	N22	5
D298	N09	N23	9
D299	N09	N24	1
D300	N09	N25	8
D301	N09	N26	2
D302	N09	N27	8
D303	N09	N28	4
D304	N09	N29	8
D305	N09	N30	10
D306	N09	N31	4
D307	N09	N32	5
D308	N09	N33	9
D309	N09	N34	2
D310	N09	N35	1
D311	N09	N36	2
D312	N09	N37	7
D313	N09	N38	7
D314	N09	N39	2
D315	N09	N40	4
D316	N10	N11	4
D317	N10	N12	9
D318	N10	N13	3
D319	N10	N14	6
D320	N10	N15	5
D321	N10	N16	1
D322	N10	N17	6
D323	N10	N18	7
D324	N10	N19	3
D325	N10	N20	4
D326	N10	N21	9
D327	N10	N22	1
D328	N10	N23	10
D329	N10	N24	2
D330	N10	N25	8
D331	N10	N26	2
D332	N10	N27	5
D333	N10	N28	1

DEMAND	**O**	**D**	**UNITS**
D334	N10	N29	1
D335	N10	N30	10
D336	N10	N31	6
D337	N10	N32	5
D338	N10	N33	1
D339	N10	N34	9
D340	N10	N35	4
D341	N10	N36	10
D342	N10	N37	3
D343	N10	N38	4
D344	N10	N39	5
D345	N10	N40	3
D346	N11	N12	3
D347	N11	N13	1
D348	N11	N14	10
D349	N11	N15	8
D350	N11	N16	5
D351	N11	N17	3
D352	N11	N18	10
D353	N11	N19	10
D354	N11	N20	9
D355	N11	N21	9
D356	N11	N22	5
D357	N11	N23	8
D358	N11	N24	7
D359	N11	N25	9
D360	N11	N26	7
D361	N11	N27	4
D362	N11	N28	10
D363	N11	N29	8
D364	N11	N30	1
D365	N11	N31	9
D366	N11	N32	9
D367	N11	N33	7
D368	N11	N34	6
D369	N11	N35	2
D370	N11	N36	2
D371	N11	N37	2
D372	N11	N38	2
D373	N11	N39	7
D374	N11	N40	10
D375	N12	N13	9
D376	N12	N14	7
D377	N12	N15	9
D378	N12	N16	4
D379	N12	N17	8
D380	N12	N18	2
D381	N12	N19	7
D382	N12	N20	10
D383	N12	N21	6
D384	N12	N22	1
D385	N12	N23	4
D386	N12	N24	1
D387	N12	N25	6
D388	N12	N26	4
D389	N12	N27	7
D390	N12	N28	5
D391	N12	N29	3
D392	N12	N30	8
D393	N12	N31	5
D394	N12	N32	7
D395	N12	N33	4
D396	N12	N34	2
D397	N12	N35	9
D398	N12	N36	9
D399	N12	N37	9
D400	N12	N38	7
D401	N12	N39	7
D402	N12	N40	10
D403	N13	N14	9
D404	N13	N15	10
D405	N13	N16	3
D406	N13	N17	3
D407	N13	N18	10
D408	N13	N19	10
D409	N13	N20	8
D410	N13	N21	1
D411	N13	N22	6
D412	N13	N23	8
D413	N13	N24	4
D414	N13	N25	6
D415	N13	N26	9
D416	N13	N27	5
D417	N13	N28	6
D418	N13	N29	2
D419	N13	N30	10
D420	N13	N31	1
D421	N13	N32	8
D422	N13	N33	7
D423	N13	N34	4
D424	N13	N35	2
D425	N13	N36	4
D426	N13	N37	5
D427	N13	N38	6
D428	N13	N39	4
D429	N13	N40	10
D430	N14	N15	3
D431	N14	N16	3
D432	N14	N17	9
D433	N14	N18	10
D434	N14	N19	2
D435	N14	N20	1
D436	N14	N21	4
D437	N14	N22	2
D438	N14	N23	10
D439	N14	N24	8
D440	N14	N25	9
D441	N14	N26	5
D442	N14	N27	10
D443	N14	N28	7
D444	N14	N29	2
D445	N14	N30	10
D446	N14	N31	1
D447	N14	N32	2
D448	N14	N33	1
D449	N14	N34	9
D450	N14	N35	8
D451	N14	N36	10
D452	N14	N37	3
D453	N14	N38	1
D454	N14	N39	2
D455	N14	N40	5
D456	N15	N16	10
D457	N15	N17	4
D458	N15	N18	9
D459	N15	N19	4
D460	N15	N20	8
D461	N15	N21	8
D462	N15	N22	8
D463	N15	N23	6
D464	N15	N24	4
D465	N15	N25	1
D466	N15	N26	8
D467	N15	N27	3
D468	N15	N28	9
D469	N15	N29	5
D470	N15	N30	4
D471	N15	N31	6
D472	N15	N32	4
D473	N15	N33	1
D474	N15	N34	7
D475	N15	N35	1
D476	N15	N36	7
D477	N15	N37	9
D478	N15	N38	4
D479	N15	N39	3
D480	N15	N40	7
D481	N16	N17	5
D482	N16	N18	1
D483	N16	N19	4
D484	N16	N20	5
D485	N16	N21	10
D486	N16	N22	3
D487	N16	N23	2
D488	N16	N24	7
D489	N16	N25	10
D490	N16	N26	7
D491	N16	N27	6
D492	N16	N28	3
D493	N16	N29	2
D494	N16	N30	3
D495	N16	N31	6
D496	N16	N32	7
D497	N16	N33	8
D498	N16	N34	3
D499	N16	N35	4
D500	N16	N36	8
D501	N16	N37	7
D502	N16	N38	5
D503	N16	N39	6
D504	N16	N40	2
D505	N17	N18	6
D506	N17	N19	2
D507	N17	N20	4
D508	N17	N21	4
D509	N17	N22	3
D510	N17	N23	7
D511	N17	N24	8
D512	N17	N25	7
D513	N17	N26	7

DEMAND	O	D	UNITS
D514	N17	N27	4
D515	N17	N28	9
D516	N17	N29	9
D517	N17	N30	7
D518	N17	N31	2
D519	N17	N32	2
D520	N17	N33	10
D521	N17	N34	2
D522	N17	N35	9
D523	N17	N36	1
D524	N17	N37	8
D525	N17	N38	6
D526	N17	N39	2
D527	N17	N40	4
D528	N18	N19	8
D529	N18	N20	1
D530	N18	N21	3
D531	N18	N22	8
D532	N18	N23	9
D533	N18	N24	2
D534	N18	N25	5
D535	N18	N26	6
D536	N18	N27	5
D537	N18	N28	1
D538	N18	N29	10
D539	N18	N30	10
D540	N18	N31	7
D541	N18	N32	7
D542	N18	N33	10
D543	N18	N34	7
D544	N18	N35	8
D545	N18	N36	1
D546	N18	N37	2
D547	N18	N38	8
D548	N18	N39	4
D549	N18	N40	4
D550	N19	N20	3
D551	N19	N21	7
D552	N19	N22	3
D553	N19	N23	10
D554	N19	N24	8
D555	N19	N25	10
D556	N19	N26	2
D557	N19	N27	3
D558	N19	N28	5
D559	N19	N29	6
D560	N19	N30	5
D561	N19	N31	6
D562	N19	N32	6
D563	N19	N33	4
D564	N19	N34	2
D565	N19	N35	6
D566	N19	N36	2
D567	N19	N37	8
D568	N19	N38	7
D569	N19	N39	3
D570	N19	N40	3
D571	N20	N21	9
D572	N20	N22	2
D573	N20	N23	4
D574	N20	N24	4
D575	N20	N25	8
D576	N20	N26	3
D577	N20	N27	2
D578	N20	N28	3
D579	N20	N29	1
D580	N20	N30	8
D581	N20	N31	9
D582	N20	N32	8
D583	N20	N33	7
D584	N20	N34	2
D585	N20	N35	5
D586	N20	N36	3
D587	N20	N37	2
D588	N20	N38	5
D589	N20	N39	6
D590	N20	N40	7
D591	N21	N22	8
D592	N21	N23	9
D593	N21	N24	2
D594	N21	N25	6
D595	N21	N26	9
D596	N21	N27	5
D597	N21	N28	7
D598	N21	N29	5
D599	N21	N30	8
D600	N21	N31	9
D601	N21	N32	6
D602	N21	N33	5
D603	N21	N34	6
D604	N21	N35	3
D605	N21	N36	5
D606	N21	N37	4
D607	N21	N38	6
D608	N21	N39	5
D609	N21	N40	3
D610	N22	N23	7
D611	N22	N24	2
D612	N22	N25	6
D613	N22	N26	6
D614	N22	N27	9
D615	N22	N28	10
D616	N22	N29	1
D617	N22	N30	8
D618	N22	N31	5
D619	N22	N32	4
D620	N22	N33	7
D621	N22	N34	3
D622	N22	N35	2
D623	N22	N36	9
D624	N22	N37	6
D625	N22	N38	2
D626	N22	N39	3
D627	N22	N40	8
D628	N23	N24	8
D629	N23	N25	10
D630	N23	N26	5
D631	N23	N27	3
D632	N23	N28	1
D633	N23	N29	6
D634	N23	N30	5
D635	N23	N31	10
D636	N23	N32	9
D637	N23	N33	4
D638	N23	N34	10
D639	N23	N35	6
D640	N23	N36	5
D641	N23	N37	8
D642	N23	N38	9
D643	N23	N39	5
D644	N23	N40	3
D645	N24	N25	3
D646	N24	N26	10
D647	N24	N27	3
D648	N24	N28	7
D649	N24	N29	1
D650	N24	N30	5
D651	N24	N31	1
D652	N24	N32	6
D653	N24	N33	1
D654	N24	N34	1
D655	N24	N35	4
D656	N24	N36	3
D657	N24	N37	5
D658	N24	N38	5
D659	N24	N39	4
D660	N24	N40	6
D661	N25	N26	9
D662	N25	N27	7
D663	N25	N28	6
D664	N25	N29	1
D665	N25	N30	7
D666	N25	N31	3
D667	N25	N32	6
D668	N25	N33	7
D669	N25	N34	2
D670	N25	N35	8
D671	N25	N36	6
D672	N25	N37	4
D673	N25	N38	10
D674	N25	N39	2
D675	N25	N40	6
D676	N26	N27	1
D677	N26	N28	2
D678	N26	N29	10
D679	N26	N30	10
D680	N26	N31	9
D681	N26	N32	10
D682	N26	N33	6
D683	N26	N34	4
D684	N26	N35	5
D685	N26	N36	9
D686	N26	N37	8
D687	N26	N38	6
D688	N26	N39	10
D689	N26	N40	8
D690	N27	N28	8
D691	N27	N29	6
D692	N27	N30	7
D693	N27	N31	10

DEMAND	O	D	UNITS
D694	N27	N32	1
D695	N27	N33	5
D696	N27	N34	10
D697	N27	N35	7
D698	N27	N36	7
D699	N27	N37	8
D700	N27	N38	6
D701	N27	N39	10
D702	N27	N40	8
D703	N28	N29	10
D704	N28	N30	1
D705	N28	N31	6
D706	N28	N32	1
D707	N28	N33	9
D708	N28	N34	6
D709	N28	N35	10
D710	N28	N36	9
D711	N28	N37	4
D712	N28	N38	5
D713	N28	N39	4
D714	N28	N40	3
D715	N29	N30	4
D716	N29	N31	3
D717	N29	N32	6
D718	N29	N33	3
D719	N29	N34	4
D720	N29	N35	6
D721	N29	N36	9
D722	N29	N37	10
D723	N29	N38	6
D724	N29	N39	7
D725	N29	N40	10
D726	N30	N31	4
D727	N30	N32	5
D728	N30	N33	3
D729	N30	N34	6
D730	N30	N35	1
D731	N30	N36	10
D732	N30	N37	1
D733	N30	N38	8
D734	N30	N39	6
D735	N30	N40	5
D736	N31	N32	6
D737	N31	N33	4
D738	N31	N34	3
D739	N31	N35	5
D740	N31	N36	5
D741	N31	N37	8
D742	N31	N38	4
D743	N31	N39	10
D744	N31	N40	4
D745	N32	N33	1
D746	N32	N34	2
D747	N32	N35	3
D748	N32	N36	3
D749	N32	N37	3
D750	N32	N38	4
D751	N32	N39	9
D752	N32	N40	7
D753	N33	N34	10
D754	N33	N35	7
D755	N33	N36	10
D756	N33	N37	9
D757	N33	N38	8
D758	N33	N39	4
D759	N33	N40	10
D760	N34	N35	10
D761	N34	N36	3
D762	N34	N37	8
D763	N34	N38	9
D764	N34	N39	3
D765	N34	N40	8
D766	N35	N36	1
D767	N35	N37	1
D768	N35	N38	4
D769	N35	N39	5
D770	N35	N40	1
D771	N36	N37	6
D772	N36	N38	5
D773	N36	N39	7
D774	N36	N40	3
D775	N37	N38	5
D776	N37	N39	8
D777	N37	N40	5
D778	N38	N39	10
D779	N38	N40	9
D780	N39	N40	5